STOCHASTIC LARGE-SCALE MACHINE LEARNING ALGORITHMS WITH DISTRIBUTED FEATURES AND OBSERVATIONS

IEEE Big Data 2024 Conference

Author: Biyi Fang, Diego Klabjan, Truong Vo

Engineering Science and Applied Mathematics
Industrial Engineering and Management Sciences

Northwestern | McCORMICK SCHOOL OF ENGINEERING

INTRODUCTION AND RELATED WORKS

Motivation

- Challenges:
  - Large datasets have high number of observations and high feature dimension
  - Large datasets exceed single-server capacity to store, thus observations and features are split and distributed across multiple machines/servers
- Objective
  - Develop scalable algorithms to process datasets with distributed features and observations/samples.

Related Works

- Stochastic Gradient Descent (SGD) works well with distributed observations
- Block Successive Upper-bound Minimization (BSUM) addresses distributed features
- Random Distributed Stochastic Algorithm (RADiSA) addresses both distributed features and observations

ALGORITHM (SODDA)

SODDA – Workflow:

Step 1: Sub-partitioning and feature selection

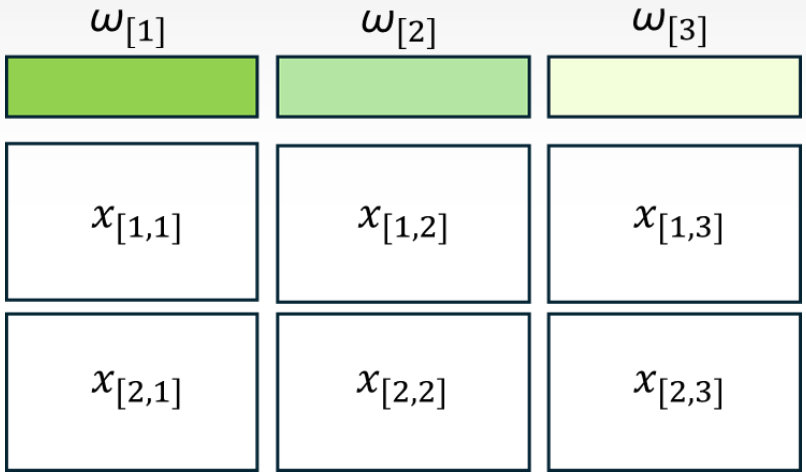
Step 2: Full-gradient estimation

Step 3: Block Parameter Update

Step 4: Aggregate Updated Parameters

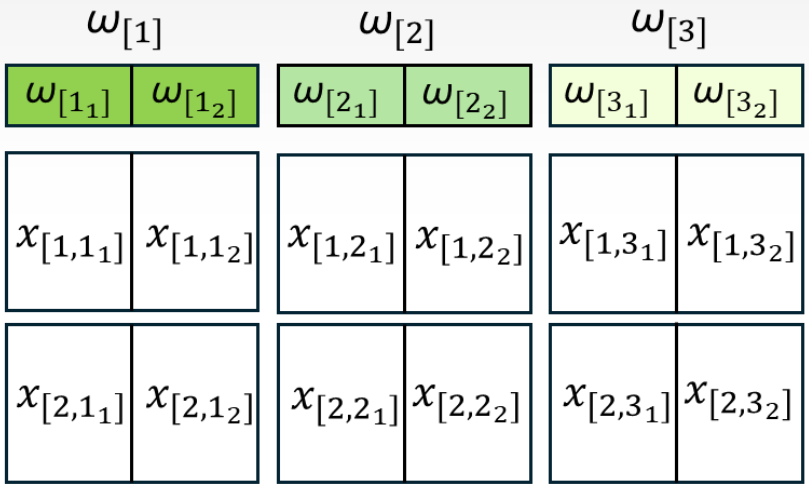
SODDA – Sub Partition:

- Given data split to Q (features) and P (observations) partitions.



SODDA – Sub Partition:

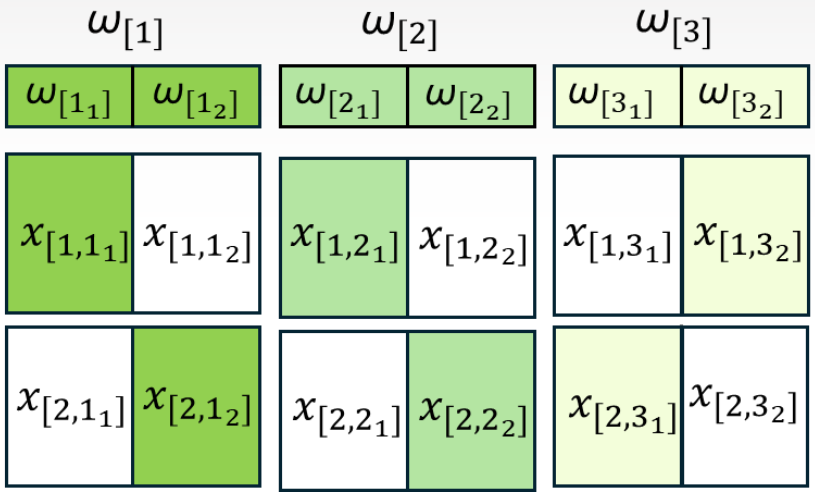
- Given data split to Q (features) and P (observations) partitions.
- Each partition is further divided to P sub-partitions with respect to ω



SODDA – Sub Partition:

- Given data split to Q (features) and P (observations) partitions.
- Each partition is further divided to P sub-partitions with respect to ω
- Select functions π_q that help define which sub-matrix are selected for partial gradient computation

$$\pi_q(p): \{1,2,\dots,p\} \rightarrow \{1,2,\dots,p\}$$



SODDA – Full Gradient Estimation:

- RADiSA computes full gradient:

$$\tilde{\mu} = \frac{1}{N} \sum_{i=1}^{N} \nabla f_i(x_i \omega^t)$$

- SODDA estimates the full gradient with partial gradient with sample subsets B_t, C_t, and D_t randomly:
  - B_t sampled from all features.
  - C_t sampled from B_t.
  - D_t sampled from all observations.

$$\mu^t = \frac{1}{d^t} \sum_{j \in D^t} \bar{\nabla}_{\omega_{ct}} f_j(x_j^{B^t} \omega_{B^t}^t)$$

SODDA – Block Parameter Update:

- Update $Q \times P$ parameter blocks in parallel, using local features and observations
- In each sub-matrix:
  - Randomly select B observations (batch size)
  - Iterate through selected samples ($i = 0\ to\ B-1$) and update corresponding parameters with SVRG

$$\bar{\omega}_q^{(i+1)} = \bar{\omega}_q^{(i)} - \gamma_{t+1}\left[\nabla_{\omega_q,\pi_q(p)} f_j^{\pi_q(p)}\left(x_j^{p,q,\pi_q(p)}\bar{\omega}_q^{(i)}\right)\right.$$

$$\left. - \nabla_{\omega_q,\pi_q(p)} f_j^{\pi_q(p)}\left(x_j^{p,q,\pi_q(p)} w_q^{(i)}\right) + \mu_q^{t,\pi_q(p)}\right]$$

SODDA –Aggregate Updated Parameters:

- Within Feature Partition q:
  - Concatenate local parameters from sub-matrices
  - Form partial parameter vector $\omega_{[q]}$

$$\omega_{[q]} = \left[\bar{\omega}_{q1}^{(B)}, \bar{\omega}_{q2}^{(B)}, \ldots, \bar{\omega}_{qP}^{(B)}\right]$$

- Across Feature Partitions:
  - Aggregate all partial solutions $\omega_{[q]}$
  - Construct complete parameter vector ω

$$\omega^{t+1} = \left[\omega_{[1]}, \omega_{[2]}, \cdots, \omega_{[Q]}\right]$$

NUMERICAL EXPERIMENTS

Experimental Datasets

- Synthetic Data

data size	small	medium	large
$P \times Q$	5×3	5×3	5×3
size of each partition	$50,000 \times 6,000$	$60,000 \times 7,000$	$60,000 \times 9,000$
Number of Spark executors used	18	25	25

Experimental Datasets

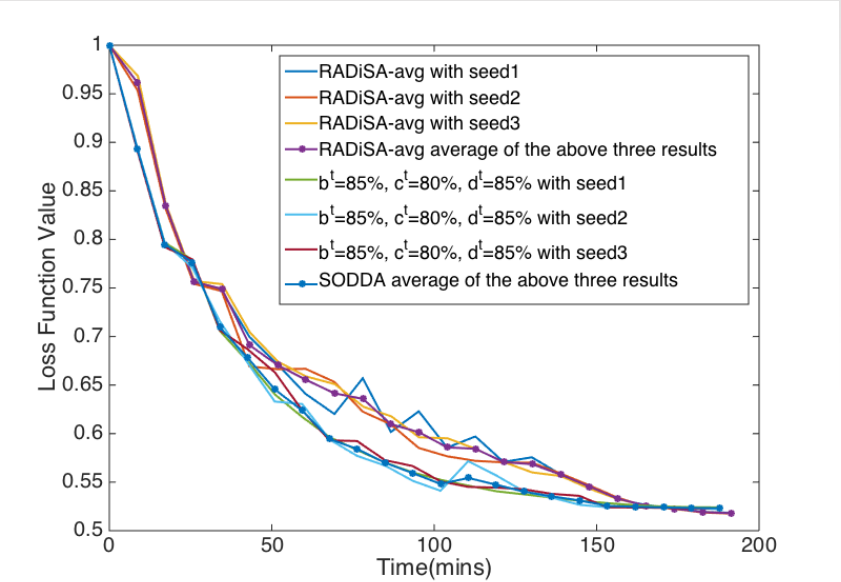
- Synthetic Data

data size	small	medium	large
$P \times Q$	5×3	5×3	5×3
size of each partition	$50,000 \times 6,000$	$60,000 \times 7,000$	$60,000 \times 9,000$
Number of Spark executors used	18	25	25

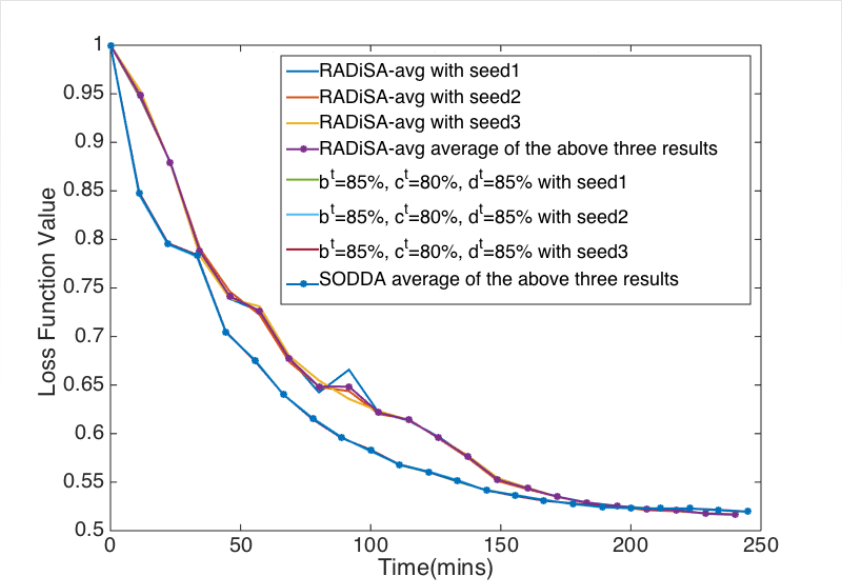
- SemMed Data

Dataset	Observations (N)	Features (d)
DIAG-neg10	425,185	26,946
LOC-neg5	5,638,696	26,966

SVM with Synthetic Data



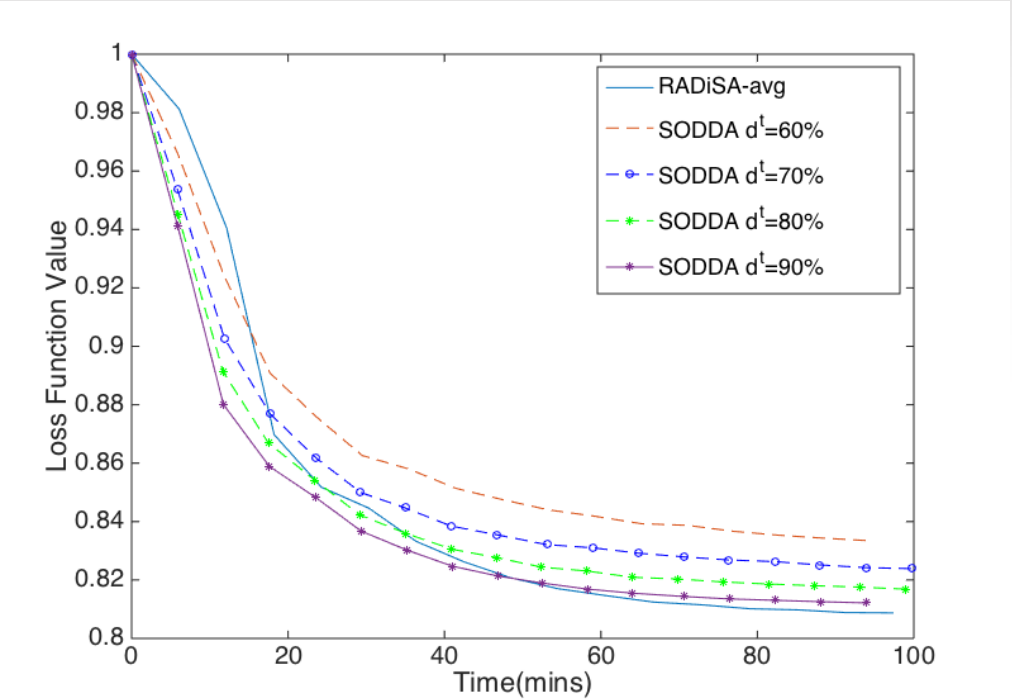
On medium dataset



On large dataset

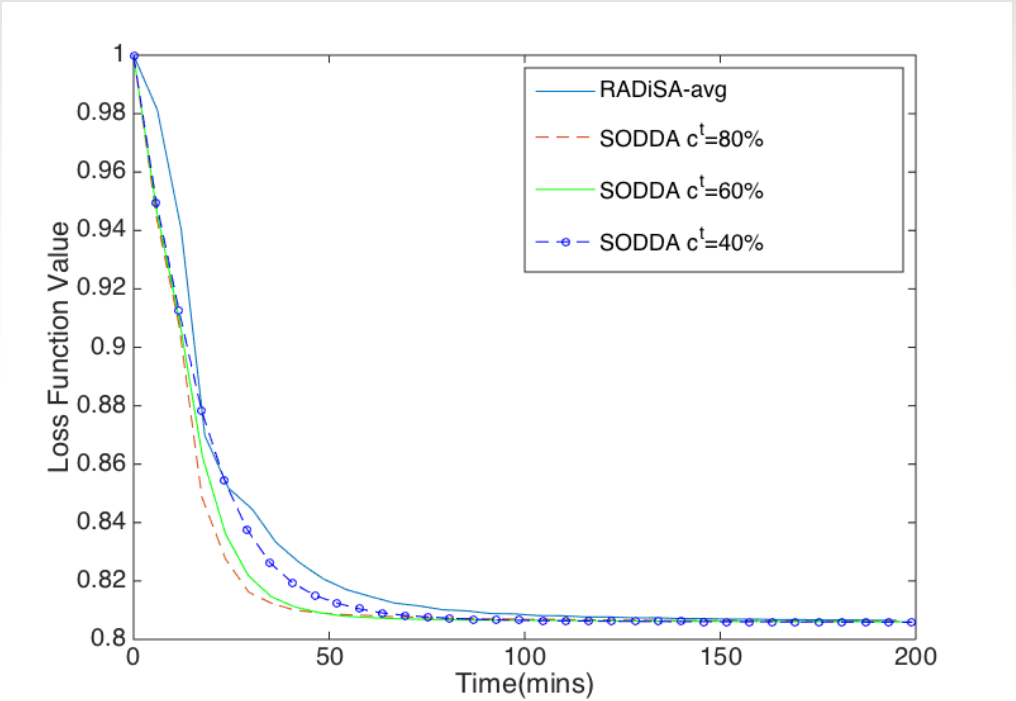
SVM with Synthetic Data

- Justification of d_t
  - Increasing from 60% to 80%: significant improvement
  - Increase from 80% to 90%: marginal benefit slowed
- Choose d_t = 85% observation



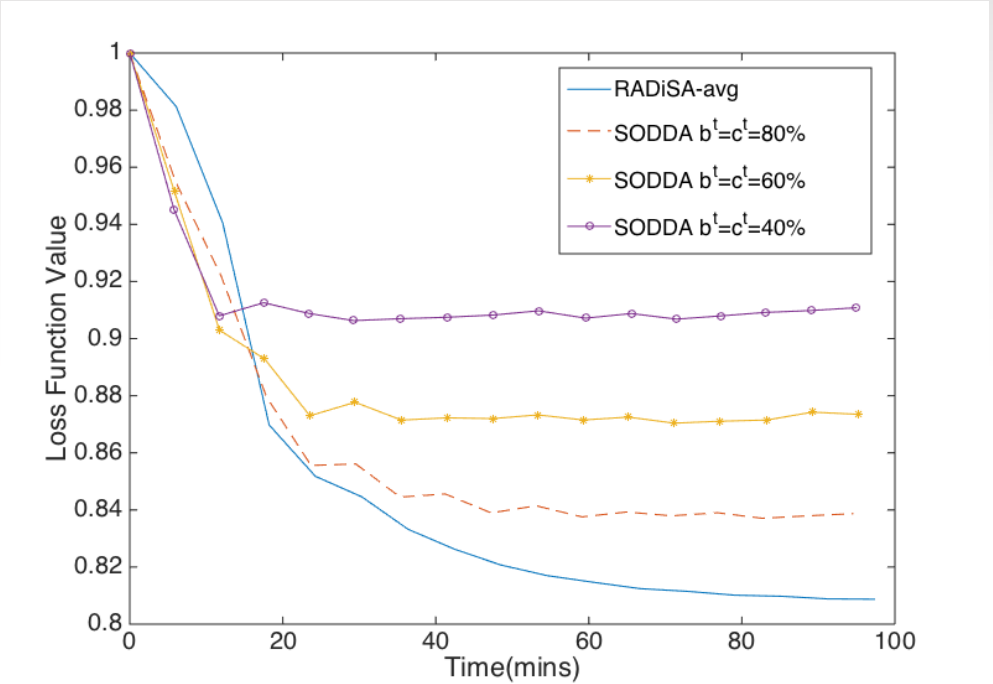
SVM with Synthetic Data

- Justification of c_t
  - Higher c_t accelerates convergence
- 80% is selected as the optimal value

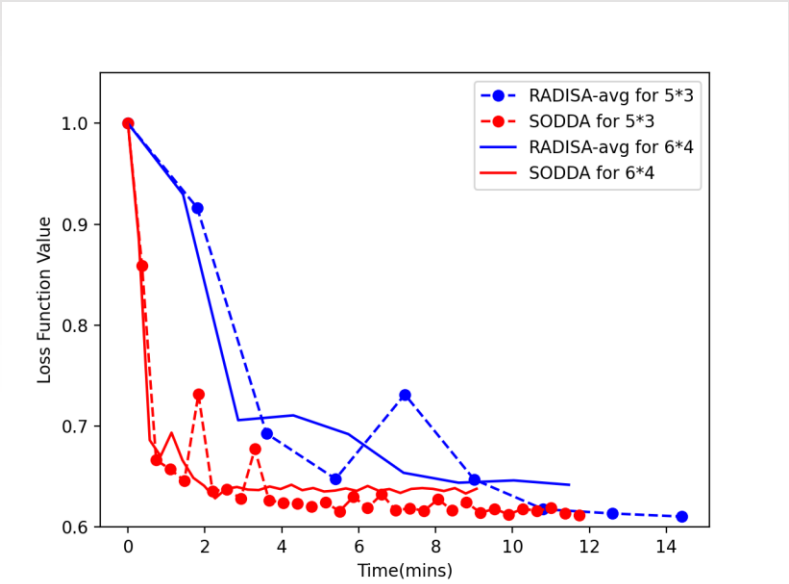


SVM with Synthetic Data

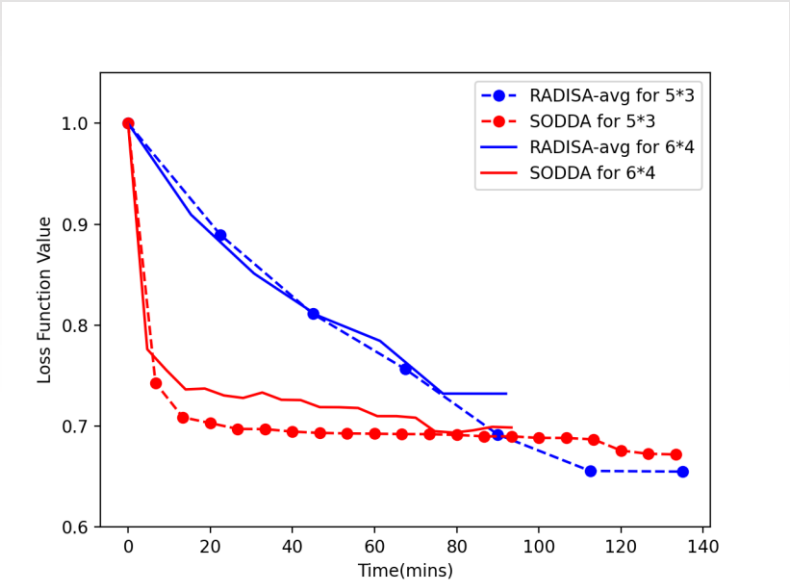
- Justification of b_t
  - Affects solution accuracy significantly.
- 85% chosen for optimal accuracy and computational time.



SVM with SemMed Databse



On DIAGNOSE dataset



On LOCATION OF dataset

Conclusion

RADiSA	SODDA
Inefficient Gradient Computation	Efficient approximated gradient
Slow Convergence rate	Stronger and faster convergence
Struggle with larger datasets	More robust to larger datasets

THANK YOU FOR YOUR ATTENTION

truongvo2025@u.northwestern.edu

Northwestern | McCORMICK SCHOOL OF ENGINEERING