Parallel Data Mining

Team 2 – Flash Coders
Team Research Investigation
Presentation 2

Foundations of Parallel Computing
Oct 2014

Agenda

- Overview of topic
- Analysis of research papers
- Software design

Overview of topic

Computational Problem

Gradient descent can take a long time to converge with many training examples, and when it does it may not have found the global optimum

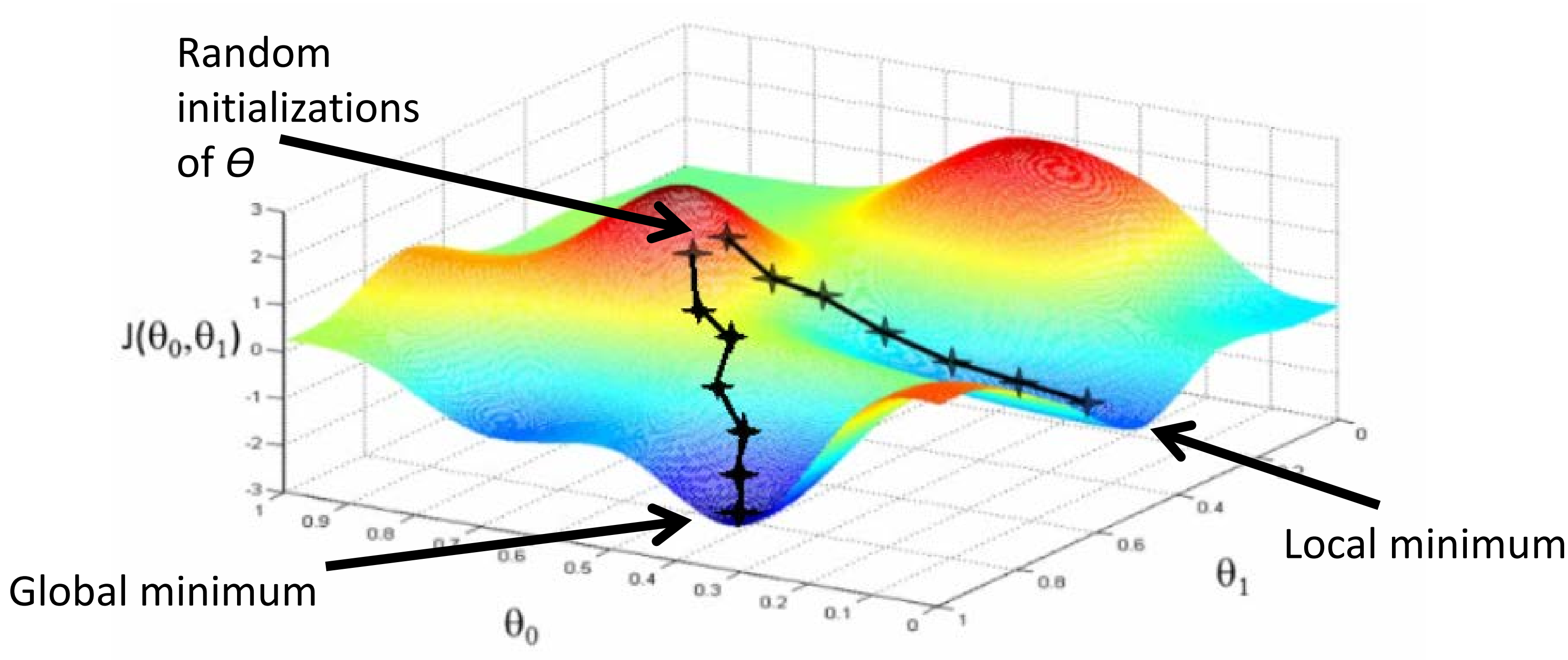


Figure from Vasilis Vryniotis. http://blog.datumbox.com/tuning-the-learning-rate-in-gradient-descent/. Accessed Sept 14, 2014.

Sequential Algorithm I

- Linear/logistic regression cost function:

What if m equals 500 million training examples!

$$\min_{\theta} J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^i) \; - y^i)^2$$

- Algorithm
  - Randomly initialize θ
  - Repeat until convergence (for all j=0,1,...,n):

$$\theta_j = \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i) {x_j}^i$$

This term is $\frac{d}{d\theta_j} J(\theta)$

Sequential Algorithm II

- Predicted output is then $h_\theta(x^i)$ with trained parameters θ
  - Linear Regression:

$$h_\theta(x^i) = \sum_{j=1}^{n} \theta^i x^i$$

  - Logistic Regression

$$z = \sum_{j=1}^{n} \theta^i x^i$$

$$h_\theta(x^i) = {}^{1}/_{(1+\ e^{-z})}$$

Paper analysis

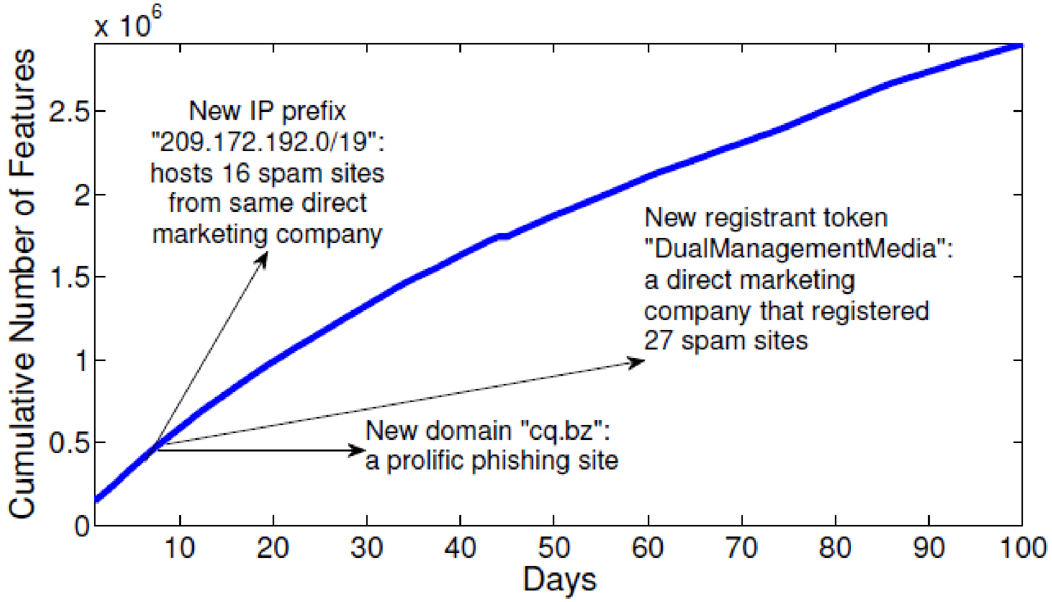
Research Paper 1

Identifying Suspicious URLs: An Application of Large-Scale Online Learning

Data collection

- Created a labeled dataset of malicious Web sites with lexical and host-based features
  - Binary BOW of URL tokens, WHOIS info, and IP data
- Predict whether site is malicious from these features only, excluding Web content
  - Faster and safer not to touch malicious content
- Used benign URLs from Yahoo directory for the non-malicious class in the dataset
- This data is also used in two of the following papers

BOW grows fast



Instances: 2396130

Attributes: 3231961

Example Instance (anonymized data):

+1 4:0.0414938 5:0.0689655 6:0.176471 26:1 54:1 56:1 62:1 64:1 66:1 68:1 70:1 72:1 933:1 4346:1 65800:1 155153:1 155154:1 155155:1 155166:1 155170:1 155174:1 155175:1 155176:1 155177:1 155178:1 155179:1 155180:1 155181:1 155182:1 155183:1 158035:1 159867:1 1284698:1 1288308:1 2112946:1 2134415:1

Results

- Used stochastic gradient descent to continually adapt to each new URL found
  - 98.4% classification rate
- Found size and velocity of dataset too large for computational complexity of batch processing
  - URL features changed too fast for the training time required
  - Model became stale if not retrained soon enough
  - A later paper will address this shortfall of batch training

Research Paper 2

Evaluating parallel logistic regression models

Problem

- Provide design guidelines to choose most effective combination of parameters for logistic regression on large data sets
- Compare and evaluate different distributed platforms, parallel algorithms, and sublinear approximation techniques.

Approaches

- Platforms:
  - Hadoop
  - Spark
- Algorithms:
  - Sequential optimization with stochastic gradient descent using LIBLINEAR (machine learning library)
  - Sublinear algorithms: use sampling of the dataset

Results I

- LIBLINEAR was the most precise and also the fastest
  - Ran on single machine
  - No inter-machine communication overhead and uses memory fully
  - Scalable, but limited by memory size
- LIBLINEAR was best if data set fits in memory, but couldn’t fit entire URL dataset
- Spark performs better than Hadoop

Results II

Hadoop vs SPARK

- Most machine learning algos iterate over the dataset
- Hadoop works off of disk while SPARK works off memory using RDDs
- Hadoop has non-negligible overhead for setting up tasks and passing parameters, especially for small datasets
- To get general sense of running time, we should consider both volume and the sparsity of data
- Observed drop in speedup with increased number of nodes in the cluster due to communication overhead (poor strong scaling)

Uses

- LIBLINEAR was the best logistic regression technique
- Our dataset, however, is too big to fit in memory. Has about 3 million features, 2 million instances, and is very sparse
- Authors recommend parallel gradient descent for this situation, which is exactly what we will do using the tardis cluster
- Measure speedups with different number of workers

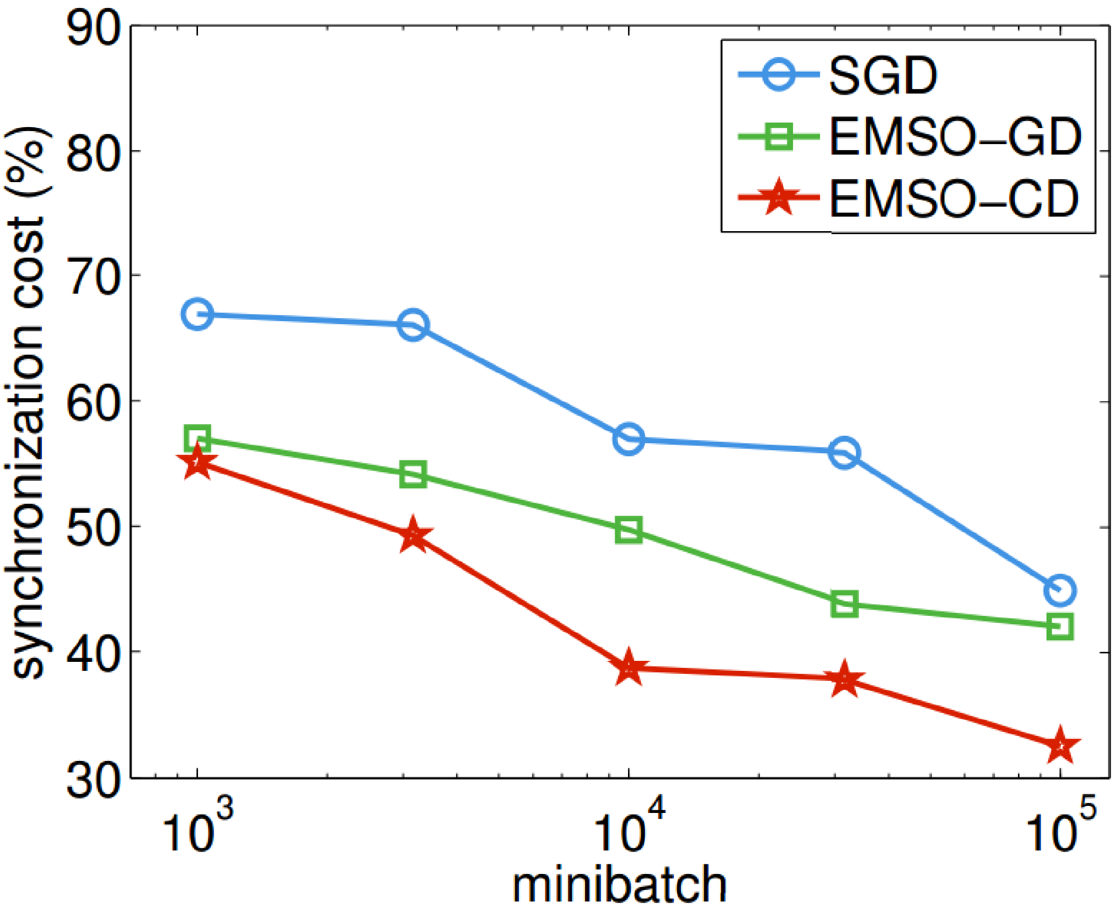
Research Paper 3

Efficient Mini-batch Training for Stochastic Optimization

Problem

- Stochastic gradient descent for large-scale optimization problems has significant communication overhead
- To reduce the communication cost in parallelized SGD, mini-batch training needs to be employed
- But, when mini-batch size increases then rate of convergence decreases (the problem in Paper 1)
  - Uses the same URL data here from Paper 1
- This paper discusses a technique that regularizes the objective function within each mini-batch iteration to prevent the convergence rate from decreasing with increased mini-batch sizes

Batching reduces synchronization cost



- The fraction of synchronization cost as a function of mini-batch size when communicating between 12 nodes
- Note cost for every algorithm decreases with batch size

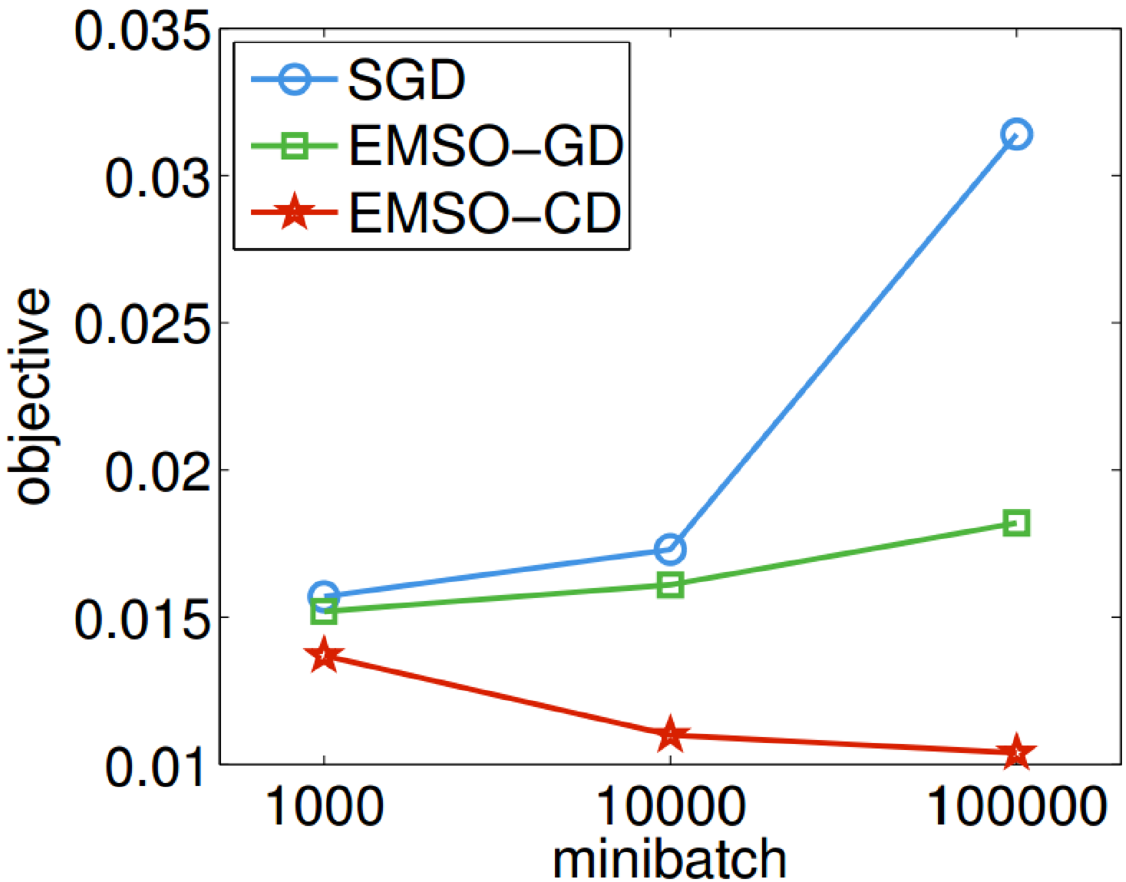
Modified minimization equation

Note: $w = \theta$

$$w_t = \underset{w \in \Omega}{\operatorname{argmin}} \left[\underbrace{\phi_{I_t}(w)}_{\text{General logistic regression}} + \underbrace{\frac{\gamma_t}{2} \|w - w_{t-1}\|_2^2}_{\text{Batch Penalty – w has larger changes in beginning and smaller at the end}} \right]$$

- Prevents divergence from previous consensus found
- Limits dramatic changes of parameter w over batch iterations

Results



- Objective value versus mini-batch size after 10^7 examples are processed
- Note SGD diverges but efficient mini-batch training does not

Use

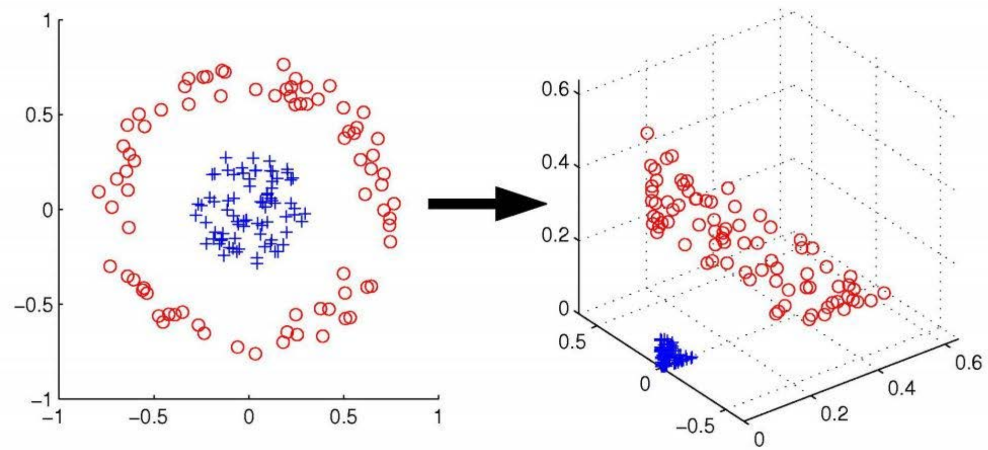
- Use the technique described for efficient mini-batch training without subsequent derease in the convergence rate

Research Paper 4

Parallel Large Scale Feature Selection for Logistic Regression

Problem

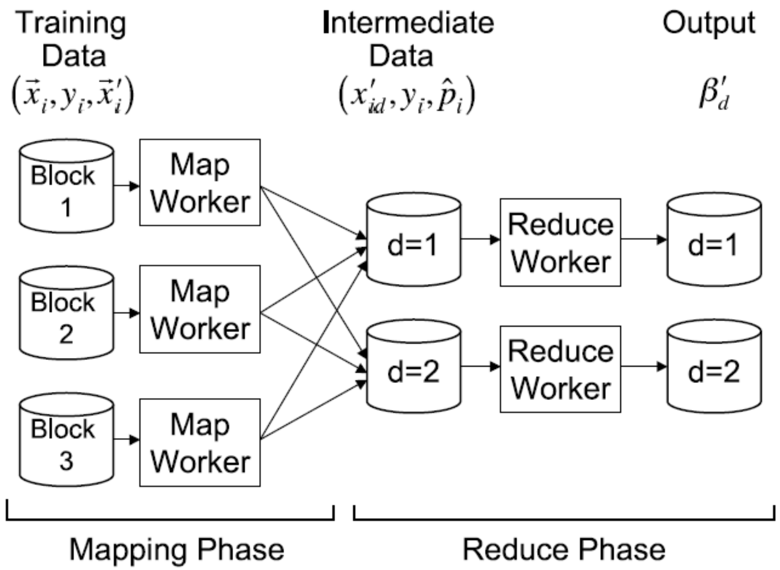
- Need to try all 2^N feature combinations to find the best model
  - In sparse, high-dimensional data this is infeasible
- Logistic regression works well in high-dimensional data since it is likely linearly separable (this is the premise of kernels)



Theory

- Approximate the best feature to add to an existing model by building a new model in parallel for each feature NOT already in the model
  - Dth iteration, need to train N – D models
- Add the best feature found to the model and re-run full logistic regression, to make sure approximation error does not propagate
  - Have to run full regression N times, or stop short

Parallel Algorithm



MapFunction$(\{\mathbf{X}, \vec{y}\}, \vec{\beta})$

Input: A data block $\{\mathbf{X}, \vec{y}\}$ and model $\vec{\beta}$.

Output: Intermediate data sets $T_d \forall d$.

FOR each $\{\vec{x}_i, y_i, \vec{x}'_i\}$ in $\{\mathbf{X}, \vec{y}\}$:

  $p_i = f(\vec{x}_i, \vec{\beta})$

  FOR each $x'_{id} \in \vec{x}'_i$:

    Store (y_i, p_i) in the intermediate data T_d

    $T_d = T_d \cup (x'_{id}, y_i, p_i)$

ReduceFunction(T_d)

Input: An intermediate data set T_d.

Output: Estimated coefficient β'_d.

$\beta'_d = 0$

Until convergence of β'_d:

  $\frac{\partial L}{\partial \beta'_d} = \frac{\partial^2 L}{\partial \beta'^2_d} = 0$

  FOR each $(x'_{id}, y_i, p_i) \in T_d$:

    $a_i = \log\left(\frac{p_i}{1-p_i}\right)$

    $p'_i = \frac{e^{a_i + \beta'_d}}{1 + e^{a_i + \beta'_d}}$

    $\frac{\partial L}{\partial \beta'_d} = \frac{\partial L}{\partial \beta'_d} + (y_i - p'_i)x'_{id}$

    $\frac{\partial^2 L}{\partial \beta'^2_d} = \frac{\partial^2 L}{\partial \beta'^2_d} - p'_i(1 - p'_i)x'^2_{id}$

  $\beta'_d = \beta'_d - \frac{\partial L}{\partial \beta'_d} / \frac{\partial^2 L}{\partial \beta'^2_d}$

Note:

- Authors used maximization of loglikelihood but this is easily mapped in minimization of squared error
- Newton's method of optimization used instead of stochastic gradient descent

Results

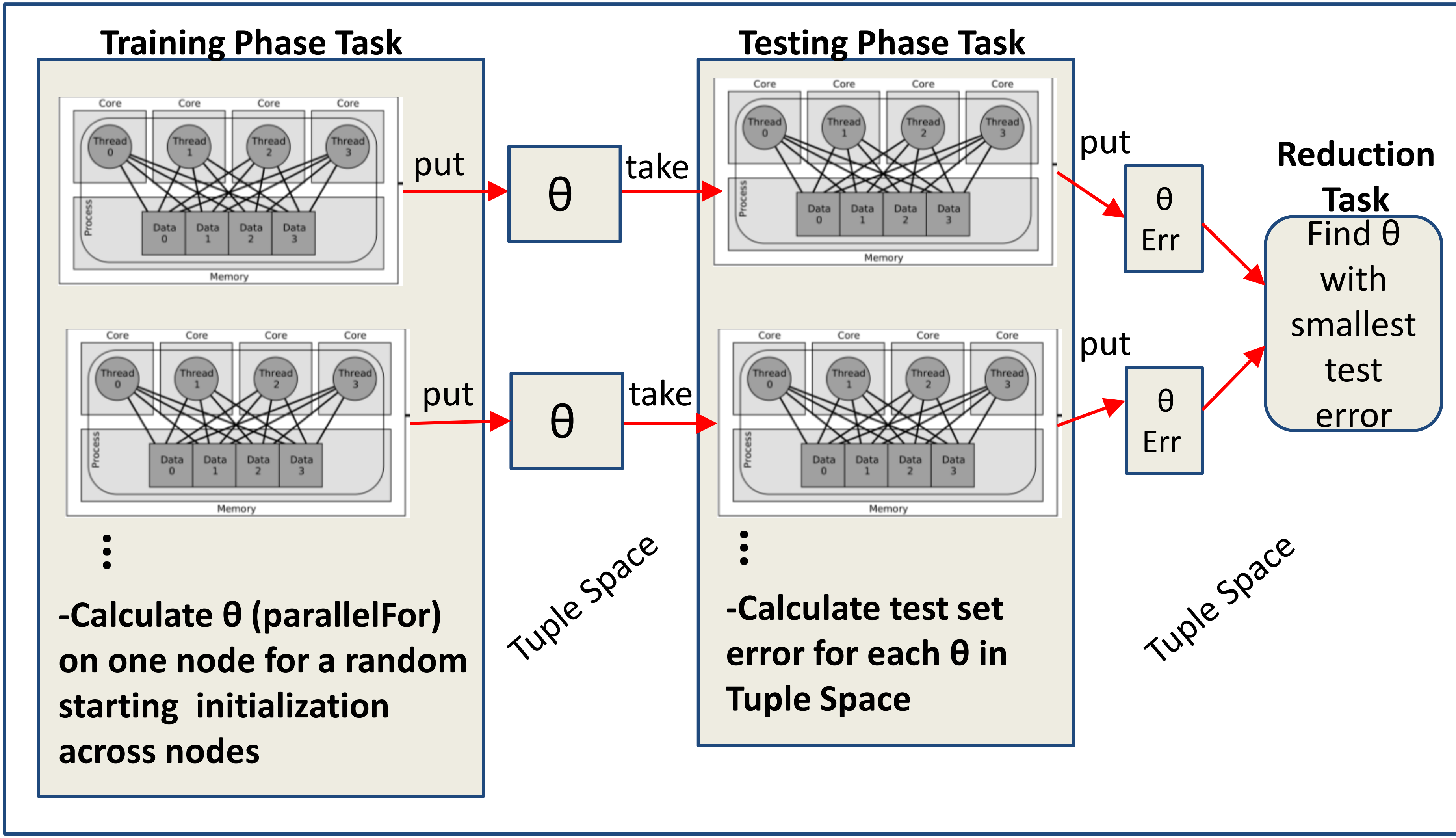
- Accurately found the true optimal feature to add on many iterations of feature selection
- Accuracy decreased as the number of irrelevant features in the dataset increased
  - Larger search space to find the true optimal
- Accuracy decreased as the number of base features increased
  - Each added feature has smaller marginal effect, making optimal one harder to find
- Feature rankings stable after looking at only 28% of the data

Use

- To improve our model, can use feature selection to grab only those relevant features of the 3 million inside the URL data
  - Good performance and comparable to results found in re-learning the whole model

Software design

JOB



Images from BIG CPU, BIG DATA: Solving the World's Toughest Computational Problems with Parallel Computing. Alan Kaminsky, 2014.

References

- Sameer Singh, Jeremy Kubica, Scott Larsen and Daria Sorokina. 2009. Parallel Large Scale Feature Selection for Logistic Regression. In *Proceedings of 2009 Society for Industrial and Applied Mathematics* (SIAM) Data Mining, SIAM, Philadelphia, PA, USA, 1172-1183. **URL**:http://epubs.siam.org/doi/pdf/10.1137/1.9781611972795.100
- Haoruo Peng, Ding Liang, and C. Choi. Evaluating parallel logistic regression models. In *Proceedings of the 2013 IEEE International Conference on Big Data*, pp 119-126, 6-9 Oct 2013. **URL**:http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6691743&isnumber=6690588
- Mu Li, Tong Zhang, Yuqiang Chen, and Alexander J. Smola. 2014. Efficient mini-batch training for stochastic optimization. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (KDD '14). ACM, New York, NY, USA, 661-670. **URL**:http://dl.acm.org/citation.cfm?id=2623330.2623612&coll=DL&dl=ACM&CFID=415399891&CFTOKEN=69514427
- Justin Ma, Lawrence K. Saul, Stefan Savage, and Geoffrey M. Voelker. 2009. Identifying suspicious URLs: an application of large-scale online learning. In *Proceedings of the 26th Annual International Conference on Machine Learning* (ICML '09). ACM, New York, NY, USA, 681-688. **URL**:http://cseweb.ucsd.edu/~savage/papers/ICML09.pdf

Further References

- Michele Banko and Eric Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics* (ACL '01). Association for Computational Linguistics, Stroudsburg, PA, USA, 26-33. **URL**: http://dl.acm.org/citation.cfm?id=1073017
- Mohammed Javeed Zaki. 1999. Parallel and Distributed Data Mining: An Introduction. In *Revised Papers from Large-Scale Parallel Data Mining, Workshop on Large-Scale Parallel KDD Systems, SIGKDD*, Mohammed Javeed Zaki and Ching-Tien Ho (Eds.). Springer-Verlag, London, UK, UK, 1-23. **URL**: http://dl.acm.org/citation.cfm?id=744383
- Andrew Ng. Machine Learning course materials, Coursera. https://www.coursera.org/course/ml. Accessed September 10, 2014.

Questions