Distributed Linear Programming Boost

Nilesh Kulkarni (110050007)
Under the guidance of
Prof. Ganesh Ramakrishnan

October 8, 2016

LPBoost

- Supervised Learning Classifier
- Iterative algorithm
- Final Learned classifier has following form $f(x) = \sum_{j=1}^{J} \alpha_j h_j(x)$
- α represent the weights for individual classifier
- LPBoost solves a linear program with infinite variables $O(2^n)$

LPBoost

Primal

$$\min_{\alpha,\zeta,\rho} -\rho + D\sum_{n=1}^{l}\zeta_n$$

$$\text{s.b.t} \sum_{h_i \in H} y_n\alpha_i h_i + \zeta_n \geq \rho,$$

$$n = 1 \dots l,$$

$$\sum_{i \in H}\alpha_i = 1,$$

$$\zeta_n \geq 0, \quad n = 1, \dots, l$$

$$\alpha_i \geq 0, \quad h_i \in H \tag{1}$$

Dual

$$\arg\min_{\mu,\beta} \beta$$

$$\text{s.b.t} \sum_{i=1}^{l}\Omega_i H_{ij} \leq \beta \; j = 1 \dots m$$

$$\sum_{i=1}^{l}\Omega_i = 1, \; 0 \leq \Omega_i \leq D \quad i = 1 \tag{2}$$

LPBoost I
Algo

Procedure 1 Algorithm For LPBoost

```
1: Given as input training set: $S$
2: $n \leftarrow 0$ No of weak hypotheses
3: $a \leftarrow 0$ All coefficients are 0
4: $\beta \leftarrow 0$
5: $u \leftarrow (\frac{1}{m}, \cdots, \frac{1}{m})$ Corresponding optimal dual
6: repeat
7:     $n \leftarrow n + 1$
8:     Find weak hypothesis by
9:         $$\sum_{i=1}^{m} y_i \hat{h}(x_i) \hat{u}_i = \max_{h \in \mathcal{H}} \sum_{i=1}^{m} \hat{u}_i y_i h(x_i)$$
10:    $h_n \leftarrow \mathcal{H}(S, u)$
```

LPBoost II
Algo

```
11:     if $\sum_{i=1}^{m} u_i y_i h_n(x_i) \leq \beta$ then
12:         $n \leftarrow n - 1$
13:         break
14:     end if
15:     $H_{in} \leftarrow h_n(x_i)$
16:     get $u, \beta$ by solving the optimization problem 2
17: until FOR EVER
18: $a \leftarrow$ Lagrangian multipliers from last LP
19: return $n, f = \sum_{j=1}^{n} a_j h_j$
```

Motivation

- hypothesis space get huge in orders of exponential
- Data is at different nodes
- Motivated from where it discusses task and data distribution on a Map-Reduce framework to solve ILPs
- A distributed solution that still preserves the sparsity of LPBoost in the hypothesis space

Distributed LPBoost-DAL along with ADMM

DAL = Dual Augmented Lagrangian
The dual version of the lpboost problem is as follows:

$$\begin{aligned} &\arg\min_{\Omega,\beta} \beta \\ &\text{s.t} \sum_{i=1}^{l} y_i \Omega_i H_{ij} \leq \beta \; j = 1 \ldots m \\ &\sum_{i=1}^{l} \Omega_i = 1, \;\; 0 \leq \Omega_i \leq D \; i = 1 \ldots l \end{aligned} \tag{3}$$

Ω_i is the weight for sample x_i
H_{ij} corresponds to $h_j(x_i)$

- Amenable to distribution in hypothesis space
- Consensus required for β and Ω's
- K Machines, Hypothesis Space Partitioned into $P_k's$, $k = 1, \ldots, K$

Distributed LPBoost-DAL along with ADMM Cont.

The dual version of the lpboost problem is as follows:

- Variables at Masters β and μ
- Variables at workers β_k and $\mathbf{\Omega_k}$

New Constraints

$$\begin{aligned} \beta &= \beta_j \ , j = 1 \dots K \\ \mu &= \mathbf{\Omega_j} \ , j = 1 \dots K \end{aligned} \tag{4}$$

LPBoost via DAL and ADMM Cont

Now the new formulation is

$$\begin{aligned}
&\arg\min_{\beta,\mu,\beta_k,\Omega_k} \beta + \sum_{k=1}^{K} \lambda_k(\beta_k - \beta) + \sum_{k=1}^{K} \psi_{\mathbf{i}}{}^{T} \\
&(\boldsymbol{\Omega}_{\mathbf{k}} - \mu) + \frac{\rho}{2}\sum_{k=1}^{K} ||\beta - \beta_k||^2 + \frac{\rho}{2}\sum_{k=1}^{K} ||\mu - \boldsymbol{\Omega}_{\mathbf{k}}||^2 \\
&\text{s.t}\forall j \in P_k \quad \sum_{i}^{l} \Omega_{ki} y_i H_{ij} \le \beta_k \\
&\sum_{i=1}^{l} \Omega_{ki} = 1, \\
&0 \le \Omega_{ki} \le D \; i = 1 \ldots l
\end{aligned} \tag{5}$$

Update to Master Variables

$$\beta^{t+1} = \frac{\sum_{k=1}^{K}(\lambda_k^t + \rho\beta_k^t)}{K}$$
$$\mu_{\mathbf{i}}^{\mathbf{t+1}} = \frac{\sum_{k=1}^{K}(\psi_{i,k}^t + \rho\Omega_{i,k}^t)}{K} \tag{6}$$

Local Optimization Problem

$$\arg\min_{\beta_k,\mathbf{\Omega_k}} \beta_k + \lambda_k(\beta_k - \beta) + \psi_{\mathbf{k}}{}^T(\mathbf{\Omega_k} - \mu) + \frac{\rho}{2}||\beta - \beta_k||^2 + \frac{\rho}{2}||\mu - \mathbf{\Omega_k}||^2$$
$$\text{s.t}\forall j \in P_k \quad \sum_{i}^{I}\Omega_{ki}y_iH_{ij} \leq \beta_k \quad \sum_{i=1}^{I}\Omega_{ki} = 1, \quad 0 \leq \Omega_i \leq D \; i = 1\ldots I \tag{7}$$

Procedure 2 Algorithm For LPBoost

```
 1: $P_k$ be the search space of the hypothesis for the k$^{th}$ worker
 2: $H_k$ be the current in hypothesis k$^{th}$ worker; Master: $\beta$ and $\mu$
 3: loop
 4:     Compute the most violating hypothesis as
        $h_{max} = \arg\max_h \sum_{i=1}^{m} \Omega_i y_i h(x_i)$
 5:     if $\sum_{i=1}^{m} \Omega_i y_i h_{max}(x_i) \leq \beta_k + \epsilon$ then
 6:         No more hypothesis to add to $H_k$
 7:         Solve the dual model 7 with hypothesis set as $H_k$ break;
 8:     else
 9:         Add $h_{max}$ to the set $H_k$
10:         Solve the dual model 7 with hypothesis set as $H_k$
11:     end if
12: end loop
    Update dual variables $\lambda_k$ and $\psi_{\mathbf{k}}$
```

Updates to Dual Variables

$$\begin{aligned} \lambda_k^{t+1} &= \lambda_{t,k} + \rho(\beta_k - \beta) \\ \psi_{\mathbf{k}}^{\mathbf{t}} &= \psi_{\mathbf{k}}^{\mathbf{t-1}} + \rho(\mathbf{\Omega}_{\mathbf{k}}^{\mathbf{t}} - \mu^{\mathbf{t}}) \end{aligned} \tag{8}$$

Termination Criteria

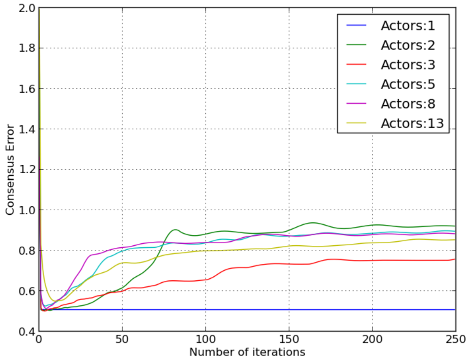
- Fixed Number of Iterations
- Duality Gap
- Relative change in Objective

Implementation

- Implemented on an event based framework Akka via *Actors*
- Currently Using fixed number of iterations as termination criteria
- Two kinds of Actors present: Master Actor and Worker Actor
- Synchronized and waits for all Worker Actors to complete jobs before moving to next iteration

Experiments

Figure: DAL-LPBoost No of Iterations vs Consensus Error



The consensus error become almost constant after some time. As we can see in the graph after 200 iterations.

Experiments Cont.

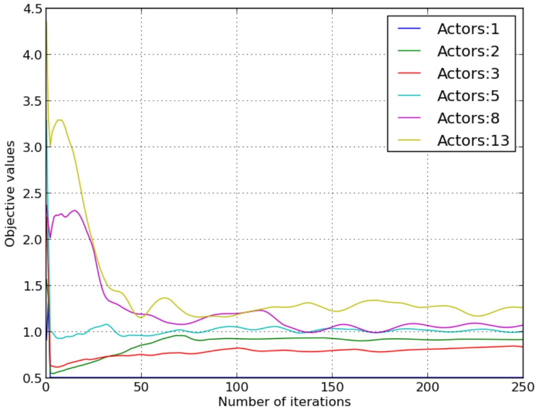
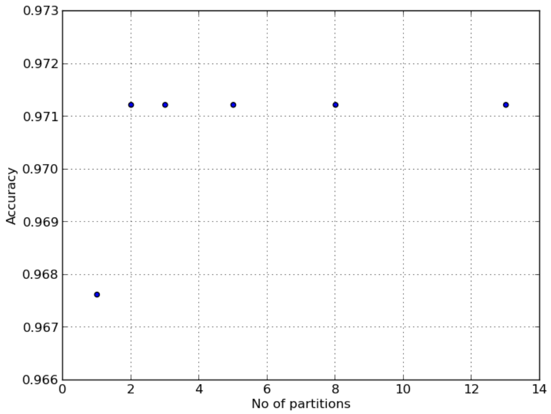


Figure: DAL-LPBoost No of Iterations vs Objective Values

The objective values all seem to converge for different number of nodes

Experiments Cont.

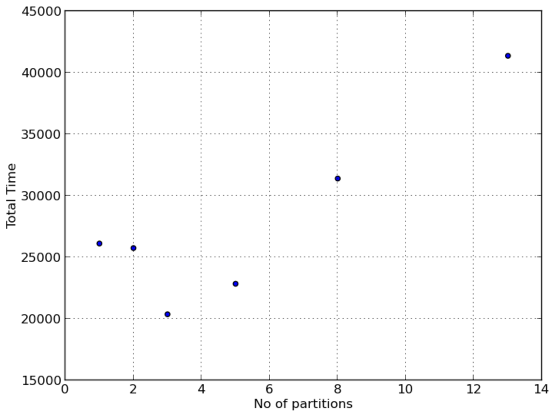
Figure: DAL-LPBoost No of Partitions vs Accuracy



The accuracy for number of nodes greater than 2 is same for all.

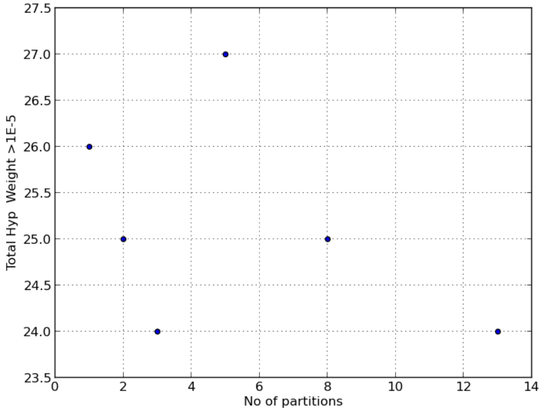
Experiments Cont.

Figure: DAL-LPBoost No of Partitions vs Time



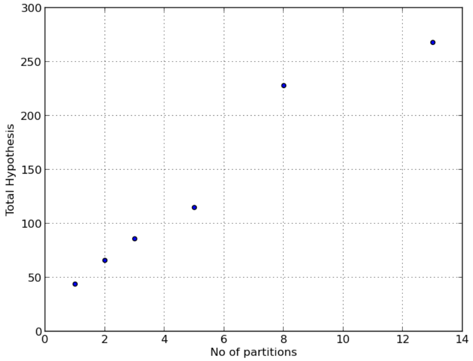
Experiments Cont.

Figure: DAL-LPBoost No of Partitions vs No of non zero Hyp (weight $\geq$ 1E-5)



Experiments Cont.

Figure: DAL-LPBoost No of Partitions vs Size of Hypothesis Set



Each node tries to operate on the data with an exclusive set of hypothesis. Hence as the number of nodes increase the Total Hypothesis sent to the master increases

As we looked at Task Distribution in the Previous few slides now look at Data Distribution. This version of LPBoost is based on the paradigm of data distribution. In contrast to the LpBoost implemented using DAL with ADMM this version is based on divide and conquer. It is motivated from the divide and conquer SVM. Data D is partitioned in sets $D_i, i = 1, \ldots, K$

Divide and Conquer LPBoost - I

Our first version of the implementation involved a very naive aggregation strategy.

Procedure 3 Divide and Conquer LPBoost Algorithm - Aggregation

```
1: K Number of partitions
2: D DataSet, $D_k$ :   $k = 1, \ldots, K$ Partitioned Data
3: Worker: Solve LPBoost problem on data partitions $D_k$ for
   every worker
4: Master: Receive weights for hypothesis($\alpha_{\mathbf{k}}$) from all the data
   partitions.
5: $\alpha \leftarrow$ Aggregate the $\alpha_{\mathbf{k}}'s$ from all the workers and report the $\alpha$,
   the final weights for all the hypothesis
```

Divide and Conquer LPBoost -II

Procedure 4 Divide and Conquer LPBoost Algorithm - Primal Based Solving at Master

```
1: K *Number of partitions*
2: D *DataSet*, $D_k$ :   $k = 1, \ldots, K$ *Partitioned Data*
3: Worker: Solve LPBoost problem on data partitions $D_k$ for every worker. $H_k$ = set of non-zeros weighing hypothesis
4: Master: Receive hypothesis set $H_k$ from all the K workers.
5: Solve for the Primal **??** using only hypothesis set which is an union of the hypothesis set received from all the worker nodes. This reduces the number of variables while solving the primal. $H = \cup_{k=1}^{K} H_k$.
```

Divide and Conquer LPBoost -III

Procedure 5 Divide and Conquer LPBoost Algorithm - Row and Column Generation

```
1: K $\textit{Number of partitions}$
2: D $\textit{DataSet}, D_k : \quad k = 1, \ldots, K$ $\textit{Partitioned Data}$
3: Worker: Solve LPBoost problem on data partitions $D_k$ for every worker. $H_k$ = set of non-zeros weighing hypothesis. $D\prime_k$ be the set of all points which have non-zero weights in the dual.
4: Master: Receive hypothesis set $H_k$ and support points $D\prime_k$ from all the K workers.
5: $H = \cup_{k=1}^{K} H_k$. $D\prime = \cup_{k=1}^{K} D\prime_k$. Solve for the Primal **??** using only hypothesis set H which is an union of the hypothesis set received from all the worker nodes along with constraints on the set of samples $\in D\prime$.
```

Experiments

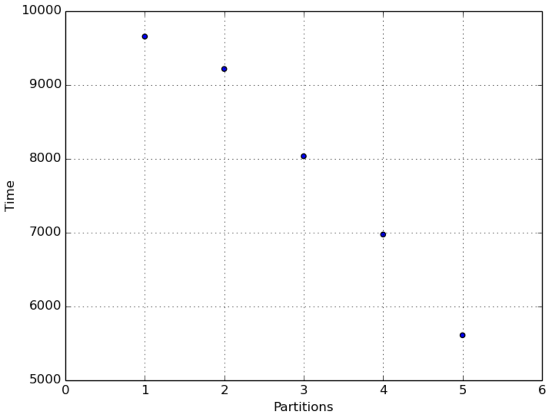
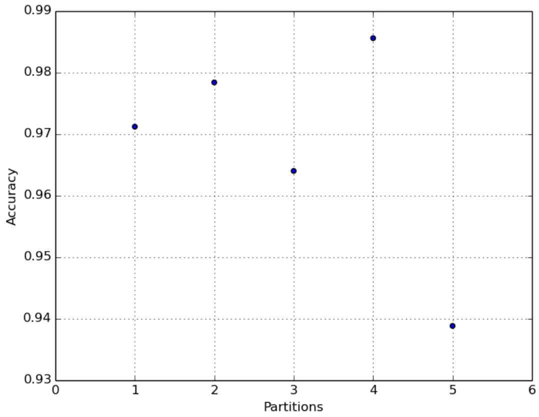


Figure: Divide and Conquer LPBoost No of Partitions vs Time

Experiments

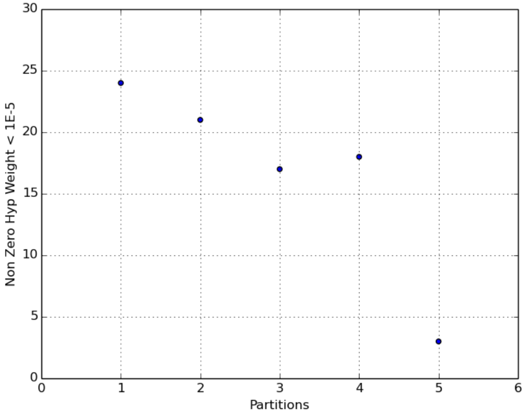
Figure: Divide and Conquer LPBoost No of Partitions vs Accuracy



The results are a bit two random because we have not run it one a

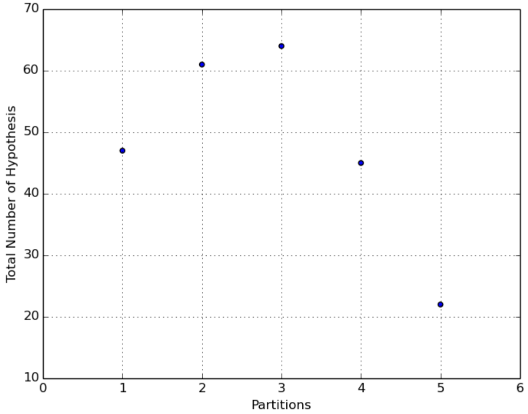
Experiments

Figure: Divide and Conquer LPBoost No of Partitions vs No of non zero Hyp (weight $\geq$ 1E-5)



Experiments

Figure: Divide and Conquer LPBoost No of Partitions vs Size of Hypothesis Set

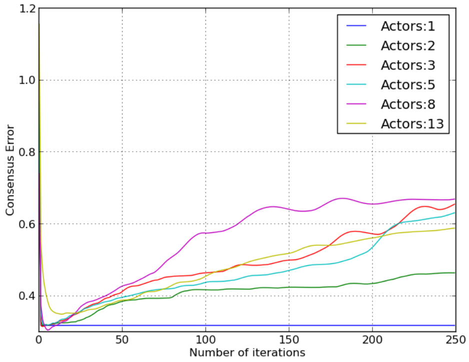


Conclusion

The algorithms needs to be tested on big datasets to check its scalability. We haven't tested it because we dont have a Cplex License under IBM Academic Initiative. Our current version supports 1000 variables and 1000 constraints. Overall we have been able to show that our implementation of Distributed LPBoost are correct and have been tested on two datasets.

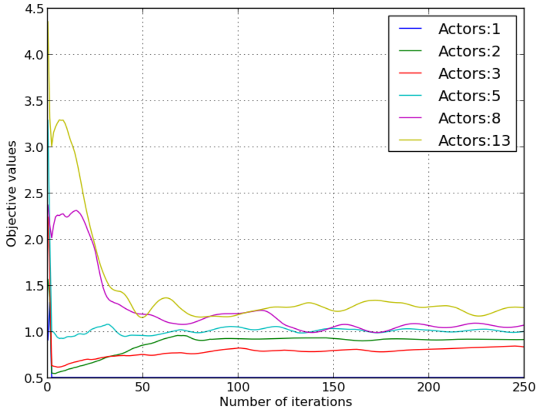
Appendix-More Graphs

Figure: DAL-LPBoost No of Iterations vs Consensus Error



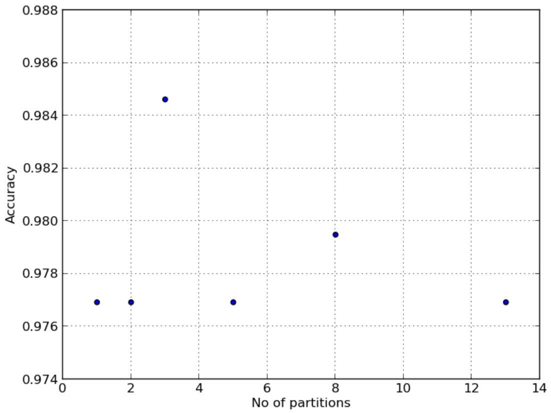
Appendix-More Graphs

Figure: DAL-LPBoost No of Iterations vs Objective Values



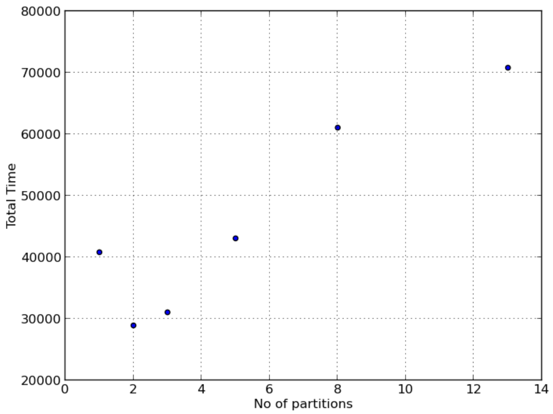
Appendix-More Graphs

Figure: DAL-LPBoost No of Partitions vs Accuracy



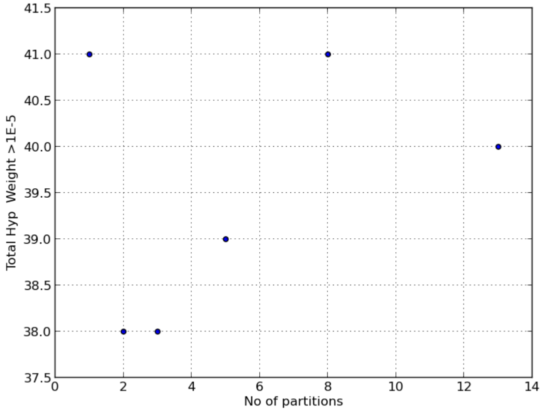
Appendix-More Graphs

Figure: DAL-LPBoost No of Partitions vs Time



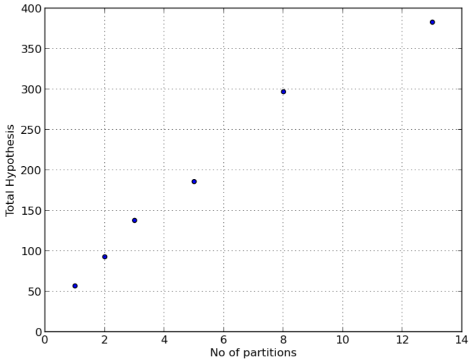
Appendix-More Graphs

Figure: DAL-LPBoost No of Partitions vs No of non zero Hyp (weight $\geq$ 1E-5)



Appendix-More Graphs

Figure: DAL-LPBoost No of Partitions vs Size of Hypothesis Set



Thank You
Questions?