Indexed Block Coordinate Descent for Large-Scale Linear Classification with Limited Memory

Ian E.H. Yen, Chun-Fu Chang, Ting-Wei Lin, Shan-Wei Lin, and Shou-De Lin
National Taiwan University

KDD 2013
Large-Scale Linear Classification

• Where is the bottleneck?
  – I/O dominates often [Yu. 2010] [Chang. 2011]
  – More serious when memory size less than data size

• Observation
  – Given large training data, usually a crucial subset of data is key to improve accuracy.
  – Referred as “dual-sparsity” in SVM literature.

• We can save a lot by reading only crucial samples into Memory
  – Challenge: Not known a priori
  – Our solution: Maintain index before Learning
Framework Overview

\[ w_1^* = \arg \min_w f_1(w; D) \quad \text{and} \quad w_k^* = \arg \min_w f_k(w; D) \]

(1) Dual-Sparse Objective

(2) Global convergent algorithm

(3) ANN indexing
Outline

• Truncated-Loss for Sublinear Dual-Sparsity
  – Sequential Relaxation for Truncated-Loss

• Indexed Learning
  – Informative sample as Nearest Neighbor
  – Indexed Block Coordinate Descent
  – Solving block sub-problem

• Implementation of Indexing

• Experiments
Improve Dual Sparsity from Linear to Sublinear --- Exploiting Truncated Loss

- Regular Support Vector Machine
  - $|SV|$ linear to $|data|$ for non-separable case

- General Truncated-Loss
  - We can modify any Convex Loss $L(.)$ to Truncated-Loss $R(.) = \min\{ L(.), 1+s \}$
  - Pros: (1) Suppress influence of outliers.
    (2) $|SV|$ sublinear to $|data|$ empirically.
  - Cons: Non-convex problem
    $\Rightarrow$ CCCP for L1-loss [Collobert. 2006]
  - General Relaxation for Truncated-Loss
Sequential Relaxation for Truncated-Loss Problem

Truncated-Loss Problem:

\[
\min_w \frac{1}{2}\|w\|^2 + C \sum_l R(y_l, w^T x_l) \quad (1)
\]

- Minimize (2) decreases objective (1).
- Reason: i. Outlier have loss \( R(.) = 1+s \), while non-outliers have loss \( R(.) = L(.) \).
  ii. Both \( 1+s \) and \( L(.) \) upper-bound \( R(.) = \min\{ L(.) , 1+s \} \).

⇒ For each iteration, ignore \( \text{OUT}(w^t) \) and solve convex-loss \( L(.) \) on only \( \text{IN}(w^t) \).

Majorization Minimization for Truncated-Loss

Initialize \( w^0 \) with convex loss \( L(.) \) learned on random sub-samples. repeat

\[
w^{t+1} = \arg\min_w \left\{ \frac{1}{2}\|w\|^2 + C \sum_{l \in \text{IN}(w^t)} L(y_l, w^T x_l) + C \sum_{l \in \text{OUT}(w^t)} 1+s \right\} \quad (2)
\]

until convergence of \( w^t \)
**Sequential Relaxation for Truncated-Loss Problem**

\[
\min_w \frac{1}{2}\|w\|^2 + C \sum_l R(y_l, w^T x_l) \tag{1}
\]

---

**Theorem 1:** The sequence \( \{w^t\}_{t=0}^\infty \) produced by (2) converges to a stationary point of (1) with at least linear rate.

**Proof:** By reduction to Block Coordinate Descent on non-convex quadratic problem:

\[
\begin{align*}
&\min_{w,\xi,\tilde{d}} \frac{1}{2}\|w\|^2 + C \sum_{l \in D} d_l \xi_l + (1 - d_l)(1 + s) \\
&\text{s.t. } y_l w^T x_l \leq 1 - \xi_l \\
&\xi_l \geq 0 \\
&0 \leq d_l \leq 1, \quad l = 1..m
\end{align*}
\]

between \( w \) and \( \mathbf{d} \).
Outline

• Truncated-Loss for Sublinear Dual-Sparsity
  – Sequential Relaxation for Truncated-Loss

• Indexed Learning
  – Informative sample as Nearest Neighbor
  – Indexed Block Coordinate Descent
  – Solving block sub-problem

• Implementation of Indexing

• Experiments
Block Coordinate Descent (BCD) in the Dual

Now we focus on L1-loss, L2-loss SVM problems:

\[
w^{t+1} = \arg \min_w \left\{ \frac{1}{2} \|w\|^2 + C \sum_{l \in \text{IN}(w^t)} L(y_l, w^T x_l) + C \sum_{l \in \text{OUT}(w^t)} 1 + s \right\}
\]

Block Coordinate Descent on the Dual:

\[
\min_{\alpha \in \mathbb{R}^N} \quad f(\alpha) = \frac{1}{2} \alpha^T \bar{Q} \alpha - e^T \alpha
\]

s.t. \quad 0 \leq \alpha_l \leq U, \quad l \in \text{IN}(w^t)

\quad \alpha_l = 0, \quad l \in \text{OUT}(w^t)

**Dual-Sparsity:** The optimal solution \( \alpha^* \) contains only \( |SV| \ll N \) non-zeros.

**Shrinking:** Iteratively eliminate non-active \( \alpha_l \) from working set. [Joachims, 1998]

To avoid I/O in limited-memory case:

**Caching:** Read partition of data into memory, caching samples with active \( \alpha_l \). [Chang, 2011]

**Indexing:** Read only samples with most active \( \alpha_l \) into memory via ANN Search Index.
Informative Samples as Nearest Neighbors

Samples with non-zero $\nabla_i^P f(\alpha)$:

\[ \{ i | L^{-1}(1 + s) \leq y_i \mathbf{w}^T \mathbf{x}_i \leq 1 \} \]

Standard ANN (similarity) search finds:

\[ \text{argmax}_i \hat{\mathbf{q}}^T \hat{\mathbf{x}}_i \]

Transform target samples as Nearest Neighbor in embedded space defined by $V(.)$:

\[ \text{argmin}_i |\hat{\mathbf{w}}^T \hat{\mathbf{x}}_i| = \text{argmin}_i (\hat{\mathbf{w}}^T \hat{\mathbf{x}}_i)^2 = \text{argmin}_i V(\hat{\mathbf{w}})^T V(\hat{\mathbf{x}}_i) = \text{argmax}_i (-V(\hat{\mathbf{w}}))^T V(\hat{\mathbf{x}}_i) \]

where $V(.)$ is degree-2 polynomial feature expansion.

$\Rightarrow$ Indexing data with product defined by $(\hat{\mathbf{x}}_i^T \hat{\mathbf{x}}_j)^2$, query with $-(\hat{\mathbf{w}}^T \hat{\mathbf{x}}_i)^2$. 
Indexed Block Coordinate Descent

**Algorithm 1 Indexed Block Coordinate Descent**

**Input:** \( w^{(t,0)} = w^t, S^{(0)} = S^t \setminus OUT(w^t) \)

**Output:** \( w^{t+1} = w^{(t,k)}, S^{t+1} = S^{(k)} \)

**repeat**

\((N, n_e) \leftarrow \text{queryIndex}( w^{(k)}; w^t, s, n ) \)

\( B \leftarrow [ N, S^{(k)}[ r : r + n_e - |N| ] ] \)

Solve block minimization problem (11) over \( B \).

\( S^{(k+1)} \leftarrow [ S^{(k)}, N ] \)

\( k \leftarrow k + 1; r \leftarrow (r + n_e + 1) \mod |S^{(k)}| \)

**until** problem (13) defined on \( S^{(k)} \) reach \( \epsilon_S \) and \( |N| < n_e \)

Cost each iteration:

\[ T_{search}(n_e) + T_{opt}(|B|), \quad n_e = \frac{n}{\text{prec}[n]} \]

Balance between these two terms:

Set \( |B| = n_c = \# \text{ of explored.} \)

Global convergence to the solution of the Dual Problem.

---

**Block Coordinate Descent**

\[
\min_{\alpha_B} f(\alpha_B; \alpha_B^{(t,k)}) \\
\text{s.t.} \quad 0 \leq \alpha_l \leq U, \quad \forall l \in B \\
\quad \quad \quad \quad \alpha_l = \alpha_l^{(t,k)}, \quad \forall l \in B
\]

**Nearest Neighbor Index**

- Reduction to ANN search
- \( n \) samples with large \( \nabla_i P f(\alpha) \)

- Active Set \( S^i \)
- Cyclic reader
- \( v.pv_1, v.pv_2, v.pv_3 \)

- dist(\( q, N(v.pv_1) \))
- dist(\( q, N(v.pv_2) \))
Solving Block Sub-problems

Block Sub-problem (Dual)

\[
\min_{\alpha_B} f(\alpha_B; \alpha_B^{(t,k)})
\]
\[
\text{s.t. } 0 \leq \alpha_l \leq U, \quad l \in B
\]
\[
\alpha_l = \alpha_l^{(t,k)}, \quad l \in \tilde{B}
\]

Block Sub-problem (Primal)

\[
\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{l \in B} L(y_l \mathbf{w}^T \mathbf{x}_l) - \mathbf{w}^T \mathbf{v}_B
\]
\[
\mathbf{v}_B = \sum_{l \in \tilde{B}} \alpha_l^{(t,k)} y_l \mathbf{x}_l
\]

They are standard Linear SVM problems. In this work, we employ:

- L1 (hinge) loss \(\Rightarrow\) Dual Coordinate Descent (DCD). [Heish, 2008]
- L2-Loss \(\Rightarrow\) Trust-Region Quasi-Newton [Lin. 2008] and DCD.
Outline

• Truncated-Loss for Sublinear Dual-Sparsity
  – Sequential Relaxation for Truncated-Loss

• Indexed Learning
  – Informative sample as Nearest Neighbor
  – Indexed Block Coordinate Descent
  – Solving block sub-problem

• Implementation of Indexing

• Experiments
Implementation of Indexing

• K-way Metric Tree
  – K reference points partition data into K subsets.
  – Recursively partitioning.

• Bias Reduction
  – Avoid bias to few reference points.
  – Bootstrap and build index for each random subsets.

• Incremental Search
  – Best-Bin-First search on each tree.
  – Traverse different trees in random order.
Outline

• Truncated-Loss for Sublinear Dual-Sparsity
  – Sequential Relaxation for Truncated-Loss

• Indexed Learning
  – Informative sample as Nearest Neighbor
  – Indexed Block Coordinate Descent
  – Solving block sub-problem

• Implementation of Indexing

• Experiments
Experiment – Data and Index

Methods Compared:

- Convex-Loss Solver
- Truncated-Loss Solver
- Indexed Truncated-Loss Solver

Solvers: (Liblinear, Pegasos)

- L1-Loss (hinge-loss)
  - Dual Coordinate Descent
  - SGD (online)

- L2-Loss
  - Trust-Region Quasi-Newton
  - Dual Coordinate Descent
  - SGD (online)

Table 1: Statistics of Data.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Samples</th>
<th>#Features</th>
<th>Storage (KB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Covtype</td>
<td>581,012</td>
<td>54</td>
<td>69,516</td>
</tr>
<tr>
<td>KDDCUP1999</td>
<td>4,898,431</td>
<td>126</td>
<td>725,180</td>
</tr>
<tr>
<td>PAMAP</td>
<td>3,850,505</td>
<td>104</td>
<td>2,198,880</td>
</tr>
<tr>
<td>Mnist8m</td>
<td>8,100,000</td>
<td>784</td>
<td>19,042,640</td>
</tr>
</tbody>
</table>

Table 2: Statistics of Index.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Storage (KB)</th>
<th>Tree Size</th>
<th>Tree Width</th>
<th>Build Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Covtype</td>
<td>446,444</td>
<td>2,000</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>KDDCUP1999</td>
<td>1,476,580</td>
<td>100,000</td>
<td>100</td>
<td>163</td>
</tr>
<tr>
<td>PAMAP</td>
<td>4,554,208</td>
<td>100,000</td>
<td>10</td>
<td>301</td>
</tr>
<tr>
<td>Mnist8m</td>
<td>20,704,784</td>
<td>10,000</td>
<td>10</td>
<td>1,539</td>
</tr>
</tbody>
</table>

• Feature scaled to [0,1].
• C=1, s=1 for all experiments.
$|SV|$ vs. Truncated-Loss parameter $(1+s)$
Testing Error vs. Time (log-scale)

L1-Loss, Dual Coordinate Descent
Testing Error vs. Time (log-scale)

**L2-Loss, Primal Trust-Region Quasi-Newton**

**L2-Loss, Dual Coordinate Descent**
$C=0.01, 0.1, 10 \text{ and } 100$

L1-Loss, Dual Coordinate Descent
Conclusion

• The bottleneck of large-scale Linear Classification lies on time spent on disk/network I/O.

• In this work, we propose Indexed Block Coordinate Descent to solve Truncated-Loss SVM with both sublinear I/O and computation time.

• Our experiments show orders of magnitude speed up when one pre-built indexing structure to help solving optimization problem.

• This is especially useful when memory is limited, or there are lots of models (from different classes, parameters, or features) to be trained.
Thank You