Efficient Projection Algorithms onto the $L_1$ Ball for Learning Sparse Representations from High Dimension Data

Yoram Singer
Google

BASED ON JOINT WORK WITH:
  JOHN DUCHI & TUSHAR CHANDRA (GOOGLE)
  SHAI SHALEV-SHWARTZ (TTI)
• Common approach to topic classification:
  • Select relevant features / tokens
  • Assign weights to tokens in order to achieve low classification error rate
A large collection of investments tools (stocks, bonds, ETFs, cash, options, ...)

Select a subset of the assets

Distribute investments among selected assets, not necessarily evenly
Many learning problems benefit from compact representation of the input space:
- spam classification, advertisements placement,
- web ranking, audio reconstruction, ...

Often the learning is divided into two phases:
- Find compact representation (CR)
- Build a prediction mechanism from (on top) CR
- Perform selection of features and learning a predictor simultaneously
• Support vector machine learning

\[ \arg \min_{\mathbf{w}} \sigma \| \mathbf{w} \|^2 + \frac{1}{m} \sum_{i=1}^{m} [1 - y_i (\mathbf{w} \cdot \mathbf{x}_i)]^+ \]

\text{PENALIZED EMPIRICAL RISK}

\[ \arg \min_{\mathbf{w} \in S} \sigma \| \mathbf{w} \|^2 + \mathbb{E}_{(\mathbf{x},y) \sim D} [\ell(\mathbf{w}; (\mathbf{x}, y))] \]

• Portfolio design

\[ \sum_{t=1}^{T} \log (\mathbf{w}_t \cdot \mathbf{x}_t) \quad \text{s.t.} \quad \mathbf{w}_t \in \Delta \]

\text{DOMAIN CONSTRAINED EMPIRICAL RISK}

\[ \sum_{t=1}^{T} \ell (\mathbf{w}_t; (\mathbf{x}_t, y_t)) \quad \text{s.t.} \quad \mathbf{w}_t \in S \]
\[
\min_{\mathbf{w}} L(\mathbf{w}) \quad \text{s.t. } \|\mathbf{w}\|_1 \leq z
\]

\[
\mathbf{w}_{t+1} = \Pi_X (\mathbf{w}_t - \eta_t \nabla_t L)
\]

\[
\Pi_X (\mathbf{w}) = \arg \min \{\|\mathbf{w} - \mathbf{v}\| \mid \mathbf{v} \in X\}
\]

\[
X = \{\mathbf{w} \mid \|\mathbf{w}\|_1 \leq z\}
\]
FOCUS MOSTLY ON EFFICIENT ALGORITHMS FOR EUCLIDEAN PROJECTIONS ONTO THE $\ell_1$ BALL IN HIGH DIMENSIONS
Projection onto $\ell_1$ Ball

\[ v_1 := v_1 - \theta \]
\[ v_2 := v_2 - \theta \]
Projection onto $\ell_1$ Ball

\[ v_1 := \max\{0, v_1 - \theta\} \]
\[ v_2 := \max\{0, v_2 - \theta\} \]
\[ \text{sign}(v_j) \max \{0, |v_j| - \theta\} \]
\[ \theta = \frac{v_1 + v_2 + v_4 + v_5 - z}{4} \]
• Had we known the threshold we could have found all the zero elements

• Had we known the elements that become zero we could have calculated the threshold
If $v_j < v_k$ then if after the projection the $k$'th component is zero, the $j$'th component must be zero as well.
If two feasible solutions exist with $k$ and $k+1$ non-zero elements then the solution with $k+1$ elements attains a lower loss.
• Sort vector to be projected

$$\Rightarrow \mu_1 \geq \mu_2 \geq \mu_3 \geq \ldots \geq \mu_n$$

• If $j$ is a feasible index then

$$\mu_j > \theta \Rightarrow \mu_j > \frac{1}{j} \left( \sum_{r=1}^{j} \mu_r - z \right)$$

• Number of non-zero elements $\rho$

$$\rho = \max \left\{ j : \mu_j - \frac{1}{j} \left( \sum_{r=1}^{j} \mu_r - z \right) > 0 \right\}$$
Calculating Projection (G)

\[ \begin{align*}
\rho &= 3 \\
\theta &= \frac{1}{3} \left( v_2 + v_4 + v_5 - z \right)
\end{align*} \]

\[ v_4 - (v_4 - z) > 0 \]

\[ v_5 - \frac{1}{2} (v_4 + v_5 - z) > 0 \]
• Assume we know number of elements greater than \( v_j \)

\[
\rho(v_j) = |\{v_i : v_i \geq v_j\}|
\]

• Assume we know the sum of elements greater than \( v_j \)

\[
s(v_j) = \sum_{i : v_i \geq v_j} v_i
\]

• Then, we can check in constant time the status of \( v_j \)

\[
v_j > \theta \iff v_j > \frac{1}{\rho(v_j)} (s(v_j) - z) \iff s(v_j) - \rho(v_j)v_j < z
\]

• Randomized median-like search \([O(n) \text{ instead } O(n \log(n))]\)
Working in High Dimensions

- In many applications the dimension is very high
  [ text application: 2 million tokens]
  [ web/ads data: often > $10^8$ ]

- Small number of non-zero elements in each example
  [ text application: ~ thousand tokens per document ]
  [ web/ads data: often < $10^{11}$ ]

- Online/stochastic updates only modify the weights corresponding to non-zero features in example

- Goals:
  - linear time in the number of non-zero features
  - sub-linear in the full dimension
Efficient Alg. for High Dim

• Use red-black (RB) tree to store only the non-zero components of the weight vector. Non-zero components are stored w/o global shift $\Theta_t = \sum_{s \leq t} \theta_t$

• Each online/stochastic update deletes & then inserts non-zero elements of an example in $O(k \log(n))$ time

• Store in each node of RB additional information that facilitates efficient search for “pivot” $\theta_t$

• Upon projection, removal of a whole sub-tree is performed in logarithmic time using Tarajan’s (83) algorithm for splitting RB tree
RB Tree for Efficient Proj.

- **Value**: 7 4 32
- **# Elements in Right Sub-Tree**: 5 2 11
- **Sum Elements in Right Sub-Tree**: 12 2 25

```
  7 4 32
 /   \
5  2  11 /   \
|     |   |     |
2  1  12 |   |  13
         |   |   \
         |   |   8 1 13
         |   |
         |   6
         |   |
         |   1
         |   6
```

21
**Pivot Search with RB Tree**

**Algorithm**

```plaintext
PROCEDURE PIVOTSSEARCH(\(T, v, \rho, s\))
Compute \(\hat{\rho} = \rho + r(v)\) and \(\hat{s} = s + \sigma(v)\)

IF \(\hat{s} < v + \frac{z}{\rho} \geq v\),
    IF \(v^\star > v\),
        \(\rho^\star = \hat{\rho}\)
        \(s^\star = \hat{s}\)
        ELSE
            \(\rho^\star = \frac{\hat{s}}{\rho^\star}\)
            \(s^\star = \frac{\hat{s}}{\rho^\star}\)
        ENDIF
    ENDIF
ENDPROCEDURE
```
Empirical Results

- Losses:
  - squared error
  - logistic regression (binary & multiclass)
- Datasets: synthetic, MNIST, Reuters Corpus Vol. 1
- Algorithms for comparison:
  - Specialized coordinate descent for SE (FHT’07)
  - Interior Point (IP) method with L₁ Boundary Const.
  - Mirror (entropic) descent & Exponentiated Gradient
Synthetic: Least Squares

- data matrix entries distributed $N(0, 1)$
- regressor:
  - 50% of components distributed $N(0, 1)$
  - 50% of components set to zero (50% irrelevant features)
- $N(0, 1)$ noise added to each target
- Cross validation to determine projection radius
- Stochastic gradient with learning rate $\sim 1/\sqrt{t}$
Synt.: Logistic Regression

- Instances generated as in least-squares
- Average over 100 runs (also in least squares)
- 10% label noise

Graph:
- N=M=800
- N=4000 M=6000

Axes:
- X: Approximate Flops
- Y: Approximate Flops

Legend:
- L1 – Batch
- L1 – Stoch
- IP
Learned predictor of the form

\[ k(x, j) = \sum_{i \in S} w_{ji} \sigma_{ji} K(x_i, x), \quad \sigma_{ji} = \begin{cases} 1 & \text{if } y_i = j \\ -1 & \text{otherwise.} \end{cases} \]

- **S**: support-set, found using multicass Perceptron
- 60,000 training examples, 28x28 pixel images
- Multiclass logistic regression with L1

\[
\min_w \quad \frac{1}{m} \sum_{i=1}^{m} \log \left( 1 + \sum_{r \neq y_i} e^{k(x_i, r) - k(x_i, y_i)} \right)
\]

\[ \text{s.t.} \quad \|w_j\|_1 \leq z, \quad w_j \succeq 0. \]
Results for MNIST Data

EG (Mirror Descent):

\[ w_j^{(t+1)} = \frac{w_j^{(t)} e^{-\eta_t \nabla_j (w^{(t)})}}{Z_t} \]
• 804,414 articles, 1,946,684 word bigrams

• Each article includes ~0.26% of bigrams

• Compared with Exponentiated Gradient (KW’97) [extension with positive & negative weights]

• Both algorithms used the same domain constraints

• Learning rate ~ 1/sqrt(t)
L₁ Proj. vs. EG on RCV1

![Graph showing cumulative loss comparison between L1 Proj. and EG on RCV1. The x-axis represents training examples, and the y-axis represents cumulative loss. The graph compares EG, EG with CCAT, EG with ECAT, L1, L1 with CCAT, and L1 with ECAT. The data points are marked with different symbols and markers for each comparison.](image-url)
L₁ Magic with RCV1

![Graph](image)

- **% Sparsity**
- **% of Total Features**
- **% of Total Seen**

**Legend:**
- Red line: % of Total Features
- Blue line: % of Total Seen

**Axes:**
- X-axis: Training Examples (0 to 8 x 10⁵)
- Y-axis: % Sparsity
Concluding Remarks

- Bertsekas first described Euclidean projection onto the simplex (see also [Gafni & Bertsekas, 84]) using sorting (O(n log(n)) time)

- Similar algorithms rediscovered and used as dual solvers for multiclass SVM, ranking problems (CS’01, CS’02, SS’06, Hazan’06)

- Efficient L\textsubscript{1}-like experts tracking: Herbster & Warmuth’01

- First efficient L\textsubscript{1} algorithms for high dimensional settings

- Part of my work on design, analysis, and implementation of provably correct & efficient learning algorithms for very large scale problems

- Extensions and other related work:
  - Adding hyper-box constraints, non-Euclidean projections
  - *Infusing AdaBoost with L\textsubscript{1} regularization*
  - New algorithm for L\textsubscript{1} regularization through projections