Pascal Challenge: Linear Support Vector Machines

Hsiang-Fu Yu
Department of Computer Science
National Taiwan University

Joint work with C.-J. Hsieh, K.-W. Chang, and C.-J. Lin
ICML Large Scale Learning Challenge, July 9, 2008
Outline

- Introduction
- Data Set Analysis
- Implementations
- Discussions
Outline

- Introduction
- Data Set Analysis
- Implementations
- Discussions
Linear Support Vector Machine Track

- We participate in Linear Track
- Reasons
  - Linear SVMs our recent research focus
  - No time to work on other tracks
Linear SVMs – Primal Forms

- Training instances \( \{y_i, x_i\}, i = 1, \ldots, l, \ y_i = \pm 1 \)

- Linear-SVM:
  \[
  \min_w \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi(w; x_i, y_i),
  \]

- L1-SVM:
  \[
  \xi(w; x_i, y_i) = \max(1 - y_i(w^T x_i), 0),
  \]

- L2-SVM:
  \[
  \xi(w; x_i, y_i) = (\max(1 - y_i(w^T x_i), 0))^2
  \]

- Linear track we are requested work on L1-SVM
Linear SVMs – Dual Forms

• L1-SVM: From primal-dual relationship

\[
\min_{\alpha} f(\alpha) = \frac{1}{2} \alpha^T Q \alpha - e^T \alpha
\]
subject to \( 0 \leq \alpha_i \leq C, \forall i, \)

• \( Q_{ij} = y_i y_j x_i^T x_j \)
• \( e \): vector of all ones
• L2-SVM is similar
LIBLINEAR: Library for Linear SVMs

We use LIBLINEAR for the competition

LIBLINEAR implements two optimization methods:

- Dual Coordinate Descent [Hsieh et al., 2008]
  - L1-loss SVM (dual)
  - L2-loss SVM (dual)
- Trust Region Newton Method [Lin et al., 2007]
  - Logistic regression
  - L2-loss SVM (primal)

Available at
http://www.csie.ntu.edu.tw/~cjlin/liblinear
Dual Coordinate Descent

- Very simple: minimizing one variable at a time
- While $\alpha$ not optimal
  For $i = 1, \ldots, l$

  $$\min_{\alpha_i} f(\cdots, \alpha_i, \cdots)$$

Details in [Hsieh et al., 2008] (talk yesterday)

- Very suitable for large sparse data
- Large # of instances and features (e.g., documents)
Dual Coordinate Descent (Cont.)

Less suitable if

- $C$ large
- Data not scaled to a small range (the same as using large $C$)
- # features small;

the case for almost all challenge sets

So dual coordinate descent may not be the most suitable

But we have no choice as

- Linear SVM track requires L1-SVM
- This is the only L1-SVM implementation in LIBLINEAR
Memory Difficulties

- LIBLINEAR uses sparse format
- Each feature value: 12 Bytes
  - index (int in LIBLINEAR; 4 Bytes)
  - value (double in LIBLINEAR; 8 Bytes)
- Our machine has 8GB RAM
- Organizers’ machine has 32GB RAM
  - used when testing participants’ code
- A direct use of LIBLINEAR requires > 32GB RAM
- Goal: Solve the whole set for each data
Outline

- Introduction
- Data Set Analysis
- Implementations
- Discussions
Data Set Analysis

Data sets: alpha, beta, gamma, delta
- instances: 500,000, features: 500
- Fully dense
- Memory requirement: 3G
  Fit our memory limit.

Data sets: epsilon, zeta
- instances: 500,000, features: 2K
- Fully dense
- Memory requirement: 12G
  Larger than our 8G
Data Set Analysis (Cont.)

Data set **face**
- instances: 5,469,800, features: 899
- Fully dense
- Memory requirement: about 59G

Data set **OCR**
- instances: 3,500,000, features: 1,156
- Fully dense
- Memory requirement: about 48G

Larger than organizers’ 32G machines
Data Set Analysis (Cont.)

Data set dna
- instances: 50,000,000, features: 800
- Feature value either 0 or 1
- 1 appears once every 4 features
- Density: 0.25
- Memory requirement: 120G

Data set webspam
- instances: 350,000, features: 16,609,143
- Density: ≤ 0.03% (the only sparse set)
- Memory requirement: about 15G
Outline

- Introduction
- Data Set Analysis
- Implementations
- Discussions
Modification of LIBLINEAR

Three data structures

For sparse data:
- Using float rather than double
- each feature value 8 bytes
- 4 bytes index; 4 bytes value
- webspam can be handled

For dense data:
- Storing value only for dense data (no index)
- each feature value 4 bytes
- face and OCR can be solved by 32G machine
Modification of LIBLINEAR (Cont.)

For 0/1 data:

- Storing index only for 0/1 data
- Use short rather than int
- Memory for dna reduced to 20G
With the above modifications, we are able to train **the whole set** of each problem.

The **only** group in linear track achieving this.
Implementations

Modification of LIBLINEAR (Cont.)

- dna: 50,000,000 instances
- The x-axis is the size of dataset; Other groups solve up to 1,000,000
Modification of LIBLINEAR (Cont.)

With the above three structures, all data sets can be handled on organizers’ machines (32G)

- However, we need to submit results (e.g., objective value)

- To generate results on our machines (≤ 8G), we need other ways

- For dna, we use only one bit to store value of each feature

  ex: \((10000010)_2\) represents 2 : 1  8 : 1
dna:

- 5G with this structure
- 120G with LIBLINEAR directly
- 20G with submitted code
- Dense bitwise operations slower
For face, OCR, we propose disk-level shrinking:

- Train 1M data
- Predict the rest
- Remove instances with $y_i (\text{predict value}) \geq 1.2$
- Guess these $\alpha_i = 0$ when training the whole
- Train the set of all remaining elements
- So we are able to handle this problem using $< 8G$
Stopping Condition

We are requested to find a solution satisfying

\[
\frac{|\text{primal} - \text{dual}|}{|\text{primal}|} \leq 0.01
\]

LIBLINEAR does not produce primal value
But can easily calculate an approximate primal value (called \text{primal}')\); details not shown here

\[
\frac{|\text{primal}' - \text{dual}|}{|\text{primal}'|} \leq 0.008
\]

We choose 0.008 as an approximate primal is used
Outline

- Introduction
- Data Set Analysis
- Implementations
- Discussions
Discussions

Evaluation (zeta)

- Time versus obj. ($C = 0.01$)
- Size versus time ($C = 0.01$)
- C versus time
Our code is fast when $C = 0.01$

- **zeta**: 500,000 instances
- Solved in around 10 sec.
Slow for **gamma**, delta, dna when $C = 10$

- Large feature values; **gamma**
  1: $-5.222623$  
  2: $-0.309575$  
  3: $-3.344716$
- **zeta**: time versus $C$

- Set a maximal number of iteration for large $C$
In practice,

- small $C$ is enough
  
  gamma: best CV at $C = 0.0009765625$

- $C$ should decrease as data size increases

\[
\frac{1}{2} w^T w + \frac{C}{l} \sum_{i=1}^{l} \max(1 - y_i (w^T x_i), 0)
\]
Solving Primal or Dual

- If the number of features is not large, the number of variables in the primal is less than the dual.
- Trust region Newton method in LIBLINEAR (L2-SVM primal) versus L2-SVM dual coordinate descent

\[
\begin{align*}
\text{gamma with } C = 1 & \quad \text{delta with } C = 1
\end{align*}
\]
When $C$ is large, trust region Newton faster for \textit{gamma, delta}

But Dual coordinate descent is faster when $C$ is small

Different methods for different situations
Discussion and Conclusions

- We successfully trained the whole set for each data.
- For large data, different approaches may be needed according to problem properties or parameters.