Outline

- Linear Support Vector Machines and Dual Problems
- General Empirical Risk Minimization
- Optimization for SVM (and general ERM problems)
Support Vector Machines

- SVM is a widely used classifier.
- Given:
  - Training data points $x_1, \cdots, x_n$.
  - Each $x_i \in \mathbb{R}^d$ is a feature vector.
- Consider a simple case with two classes: $y_i \in \{+1, -1\}$.
- Goal: Find a hyperplane to separate these two classes of data: if $y_i = 1$, $w^T x_i \geq 1$; if $y_i = -1$, $w^T x_i \leq -1$. 

![Graphical representation of SVM classification](image)
Support Vector Machines (hard constraints)

- Given training data $x_1, \cdots, x_n \in \mathbb{R}^d$ with labels $y_i \in \{+1, -1\}$.
- SVM primal problem (with hard constraints):

$$\min_{w, \xi} \frac{1}{2} w^T w$$

subject to

$$y_i (w^T x_i) \geq 1, \quad i = 1, \ldots, n,$$

- What if there are outliers?
Support Vector Machines

Given training data \( x_1, \ldots, x_n \in \mathbb{R}^d \) with labels \( y_i \in \{+1, -1\} \).

SVM primal problem:

\[
\min_{\mathbf{w}, \xi} \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^{n} \xi_i
\]

s.t. \( y_i (\mathbf{w}^T \mathbf{x}_i) \geq 1 - \xi_i, \ i = 1, \ldots, n, \)

\( \xi_i \geq 0 \)
Support Vector Machines

- SVM primal problem can be written as

\[
\min_w \frac{1}{2} \|w\|^2 + \sum_{i=1}^{n} \max(0, 1 - y_i w^T x_i)
\]

\[
\text{L2 regularization} \quad \text{hinge loss}
\]

- Non-differentiable when \(y_i w^T x_i = 1\) for some \(i\)

- Next, we show how to derive the **dual form** of SVM
Support Vector Machines (dual)

- Primal problem:

$$\min_{w, \xi} \frac{1}{2} \|w\|^2 + C \sum_i \xi_i$$

s.t. $y_i w^T x_i - 1 + \xi_i \geq 0$, and $\xi_i \geq 0 \quad \forall i = 1, \ldots, n$

- Equivalent to:

$$\min_{w, \xi} \max_{\alpha \geq 0, \beta \geq 0} \frac{1}{2} \|w\|^2 + C \sum_i \xi_i - \sum_i \alpha_i (y_i w^T x_i - 1 + \xi_i) - \sum_i \beta_i \xi_i$$

- Under certain condition (e.g., slater’s condition), exchanging min, max will not change the optimal solution:

$$\max_{\alpha \geq 0, \beta \geq 0} \min_{w, \xi} \frac{1}{2} \|w\|^2 + C \sum_i \xi_i - \sum_i \alpha_i (y_i w^T x_i - 1 + \xi_i) - \sum_i \beta_i \xi_i$$
Support Vector Machines (dual)

- Reorganize the equation:

\[
\max_{\alpha \geq 0, \beta \geq 0} \min_{w, \xi} \frac{1}{2} \|w\|^2 - \sum_i \alpha_i y_i w^T x_i + \sum_i \xi_i (C - \alpha_i - \beta_i) + \sum_i \alpha_i
\]

- Now, for any given \(\alpha, \beta\), the minimizer of \(w\) will satisfy

\[
\frac{\partial L}{\partial w} = w - \sum_i \alpha_i y_i x_i = 0 \Rightarrow w^* = \sum_i y_i \alpha_i x_i
\]

Also, we have \(C = \alpha_i + \beta_i\), otherwise \(\xi_i\) can make the objective function \(-\infty\)

- Substitute these two equations back we get

\[
\max_{\alpha \geq 0, \beta \geq 0, C=\alpha+\beta} -\frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i^T x_j + \sum_i \alpha_i
\]
Therefore, we get the following dual problem

$$\max_{c \geq \alpha \geq 0} \left\{ -\frac{1}{2} \alpha^T Q \alpha + e^T \alpha \right\} := D(\alpha),$$

where $Q$ is an $n$ by $n$ matrix with $Q_{ij} = y_i y_j x_i^T x_j$

Based on the derivations, we know

1. Primal minimum = dual maximum (under slater’s condition)
2. Let $\alpha^*$ be the dual solution and $w^*$ be the primal solution, we have

$$w^* = \sum_i y_i \alpha_i^* x_i$$

We can solve the dual problem instead of the primal problem.
General Empirical Risk Minimization

- **L2-regularized ERM:**
  
  \[
  \min_{\mathbf{w} \in \mathbb{R}^d} P(\mathbf{w}) := \sum_{i=1}^{n} \ell_i(\mathbf{w}^T \mathbf{x}_i) + \frac{1}{2}\|\mathbf{w}\|^2
  \]

- \(\ell_i(\cdot)\): loss function

- **Dual problem for L2-regularized ERM:**
  
  \[
  \min_{\alpha} D(\alpha) := \frac{1}{2}\|\sum_{i=1}^{n} \alpha_i \mathbf{x}_i\|^2 + \sum_{i=1}^{n} \ell^*_i(-\alpha_i),
  \]

- \(\ell^*_i(\cdot)\): conjugate of \(\ell\)

- **Primal-dual relationship:** \(\mathbf{w}^* = \sum_{i=1}^{n} \alpha_i \mathbf{x}_i\)
General Empirical Risk Minimization

- Regularized ERM:

\[
\min_{\mathbf{w} \in \mathbb{R}^d} P(\mathbf{w}) := \sum_{i=1}^{n} \ell_i(\mathbf{w}^T \mathbf{x}_i) + R(\mathbf{w})
\]

- \( \ell_i(\cdot) \): loss function
- \( R(\mathbf{w}) \): regularization

- Dual problem may have a different form (?)
Examples

- **Loss functions:**
  - Regression: $\ell_i(x_i) = (x_i - y_i)^2$
  - SVM (hinge loss): $\ell_i(x_i) = \max(1 - y_i w^T x_i, 0)$
  - Square hinge loss: $\ell_i(x_i) = \max(1 - y_i w^T x_i, 0)^2$
  - Logistic regression: $\ell_i(x_i) = \log(1 + e^{-y_i w^T x_i})$

- **Regularizations:**
  - L2-regularization: $\|w\|_2^2$
  - L1-regularization: $\|w\|_1$
  - Group Lasso: $\|w_{S_1}\|_2 + \|w_{S_2}\|_2 + \cdots + \|w_{S_k}\|_2$
  - Nuclear norm: $\|W\|_*$
# Optimization Methods for ERM

<table>
<thead>
<tr>
<th>Method</th>
<th>Smooth loss and regularization</th>
<th>Smooth loss nonsmooth regularization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gradient descent</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Proximal gradient</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>SGD</td>
<td>Yes</td>
<td>(Yes, with modification)</td>
</tr>
<tr>
<td>CD</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Newton</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Prox Newton</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

(assume non-smooth regularization is “simple”)

Non-smooth loss: generally hard, need to use subgradient
Stochastic Gradient for SVM
Stochastic Gradient

- Decompose the problem into $n$ parts:

$$f(w) = \frac{1}{n} \sum_i \left( \frac{1}{2} \|w\|^2 + nC \max(0, 1 - y_i w^T x_i) \right)$$

$$:= \frac{1}{n} \sum_i f_i(w)$$

- Stochastic Gradient (SG):

  For $t = 1, 2, \ldots$
  
  Randomly pick an index $i$

  $$w^{t+1} \leftarrow w^t - \eta_t \nabla f_i(w^t)$$

- Can be directly applied for smooth loss functions.

- But for SVM, $f_i$ is non-differentiable.
A vector $\mathbf{g}$ is a subgradient of $f$ at a point $\mathbf{x}_0$ if

$$f(\mathbf{x}) - f(\mathbf{x}_0) \geq \mathbf{g}^T(\mathbf{x} - \mathbf{x}_0) \quad \forall \mathbf{x}$$

Stochastic Subgradient descent:

For $t = 1, 2, \ldots$

Randomly pick an index $i$

$$\mathbf{w}^{t+1} \leftarrow \mathbf{w}^t - \eta_t g_i, \text{ where } g_i \text{ is a subgradient of } f_i \text{ at } \mathbf{w}^t$$
Stochastic Subgradient Method for SVM

A subgradient of $\ell_i(w) = \max(0, 1 - y_i w^T x_i)$:

\[
\begin{cases} 
  -y_i x_i & \text{if } 1 - y_i w^T x_i > 0 \\
  0 & \text{if } 1 - y_i w^T x_i < 0 \\
  0 & \text{if } 1 - y_i w^T x_i = 0 
\end{cases}
\]

Stochastic Subgradient descent for SVM:

For $t = 1, 2, \ldots$

Randomly pick an index $i$

If $y_i w^T x_i < 1$, then

$$w \leftarrow (1 - \eta_t)w + \eta_t nCy_i x_i$$

Else (if $y_i w^T x_i \geq 1$):

$$w \leftarrow (1 - \eta_t)w$$
Stochastic Subgradient Method

- Improve the time complexity when $\mathbf{x}_i$ is sparse:
  - $\mathbf{w} \leftarrow (1 - \eta t)\mathbf{w}$: $O(d)$ time complexity
  - Instead, we maintain $\mathbf{w} = a\mathbf{v}$, so this operation requires only $O(1)$ time
  - $\mathbf{w} \leftarrow \mathbf{w} - \eta n C y_i \mathbf{x}_i$: $O(n_i)$ time where $n_i$ is the number of non-zeroes in $\mathbf{x}_i$

- This algorithm was proposed in:
  
  Shalev-Shwartz et al., “Pegasos: Primal Estimated sub-GrAdient SOlver for SVM”, in ICML 2007
Dual Coordinate Descent for SVM
Dual Form of SVM

- Given training data \( \mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathbb{R}^d \) with labels \( y_i \in \{+1, -1\} \).
- SVM dual problem:

\[
\begin{align*}
\min_{0 \leq \alpha \leq C} & \quad \frac{1}{2} \mathbf{\alpha}^T Q \mathbf{\alpha} - \sum_{i=1}^{n} \alpha_i \\
\text{where } Q & \text{ is an } n \text{ by } n \text{ matrix with } Q_{ij} = y_i y_j \mathbf{x}_i^T \mathbf{x}_j.
\end{align*}
\]

- After computing the dual optimal solution \( \mathbf{\alpha}^* \), the primal solution is

\[
\mathbf{w}^* = \sum_{i=1}^{n} y_i \alpha_i^* \mathbf{x}_i
\]
Coordinate Descent Algorithm

- Stochastic Coordinate descent for solving $\min_{0 \leq \alpha \leq C} f(\alpha)$:

  For $t = 1, 2, \ldots$
    - Randomly pick an index $i$
    - Compute the optimal one-variable update:
      $$\delta^* = \arg\min_{\delta: 0 \leq \alpha_i + \delta \leq C} f(\alpha + \delta e_i)$$
  
  Update $\alpha_i \leftarrow \alpha_i + \delta^*$
Dual Coordinate Descent for SVM

- One-variable subproblem:

\[
f(\alpha + \delta e_i) = \frac{1}{2}(\alpha + \delta e_i)^T Q(\alpha + \delta e_i) - \sum_{i=1}^{n} \alpha_i - \delta
\]

\[
= \frac{1}{2} \alpha^T Q\alpha + \alpha^T Q e_i \delta + \frac{Q_{ii}}{2} \delta^2 - \sum_{i=1}^{n} \alpha_i - \delta
\]

- Compute \( \arg\min_{\delta} f(\alpha + \delta e_i) \): set gradient equals to zero

\[
(Q\alpha)_i + Q_{ii} \delta^* - 1 = 0 \quad \Rightarrow \quad \delta^* = \frac{1 - (Q\alpha)_i}{Q_{ii}}
\]

- However, we require \( 0 \leq \alpha_i + \delta \leq C \), so the optimal solution is

\[
\delta^* = \max \left( -\alpha_i, \min \left( C - \alpha_i, \frac{1 - (Q\alpha)_i}{Q_{ii}} \right) \right)
\]
Main computation: the $i$-th element of $Q\alpha$

Time complexity:
Main computation: the $i$-th element of $Q\alpha$

Time complexity:

$$\begin{align*}
(Q\alpha)_i &= \sum_{j=1}^{n} Q_{ij} \alpha_j \\
&= \sum_{j=1}^{n} y_i y_j x_i^T x_j \alpha_j \\
&= y_i \sum_{j=1}^{n} y_j \alpha_j x_j^T x_i
\end{align*}$$

Naive implementation: $O(nd)$ time.
A faster way to compute coordinate descent updates.

\[(Q\alpha)_i = y_i (\sum_{j=1}^{n} y_j \alpha_j x_j)^T x_i\]

Maintain \(w = \sum_{j=1}^{n} y_j \alpha_j x_j\) in the memory:

\[\Rightarrow O(d)\) time for computing \((Q\alpha)_i\);

\(w\) is the primal variables correspond to current dual solution \(\alpha\)!
After updating $\alpha_i \leftarrow \alpha_i + \delta^*$, we need to maintain $\mathbf{w}$:

$$
\mathbf{w} \leftarrow \mathbf{w} + \delta^* y_i \mathbf{x}_i
$$

Time complexity: $O(d)$

After convergence,

$$
\mathbf{w}^* = \sum_{i=1}^{n} \alpha_i^* y_i \mathbf{x}_i
$$

is the optimal primal solution.
Initial: $\alpha, \mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$

For $t = 1, 2, \ldots$

Randomly pick an index $i$

Compute the optimal one-variable update:

$$\delta^* = \max \left( -\alpha_i, \min \left( C - \alpha_i, \frac{1 - y_i \mathbf{w}^T \mathbf{x}_i}{Q_{ii}} \right) \right)$$

Update $\alpha_i \leftarrow \alpha_i + \delta^*$

Update $\mathbf{w} \leftarrow \mathbf{w} + \delta^* y_i \mathbf{x}_i$

(Hsieh et al., “A Dual Coordinate Descent Method for Large-scale Linear SVM”, ICML 2008)

Can be applied to general L2-regularized ERM problems (Shalev-Shwartz and Zhang, “Stochastic Dual Coordinate Ascent Methods for Regularized Loss Minimization” JMLR 2013)
Convergence of Dual Coordinate Descent

- Is the dual SVM problem strongly convex?
Is the dual SVM problem strongly convex?

\[ Q = \bar{X} \bar{X}^T \] may be low-rank

- Sublinear convergence in dual objective function value.
- Sublinear convergence in duality gap (primal obj - dual obj)
  (Shown in Shalev-Shwartz and Zhang, “Stochastic Dual Coordinate Ascent Methods for Regularized Loss Minimization”. JMLR 2013).
- Global linear convergence rate in terms of dual objective function value.
Experimental comparison

- RCV1: 677,399 training samples; 47,236 features; 49,556,258 nonzeroes in the whole dataset.

(e) L1-SVM: rcv1

(f) L2-SVM: rcv1

Time vs primal objective function value
Experimental comparison

- RCV1: 677,399 training samples; 47,236 features; 49,556,258 nonzeroes in the whole dataset.

(e) L1-SVM: rcv1
(f) L2-SVM: rcv1

Time vs prediction accuracy
Implemented in LIBLINEAR:
https://www.csie.ntu.edu.tw/~cjlin/liblinear/

Other functionalities:
- Logistic regression (L1 or L2 regularization)
- Multi-class SVM
- Support vector regression
- Cross-validation
Recent Research Topics
Asynchronous dual coordinate descent algorithm:

Each thread repeatedly performs the following updates:

For $t = 1, 2, \ldots$

Randomly pick an index $i$

Compute the optimal one-variable update:

$$\delta^*_i = \max \left( -\alpha_i, \min \left( C - \alpha_i, \frac{1 - y_i w^T x_i}{Q_{ii}} \right) \right)$$

Update $\alpha_i \leftarrow \alpha_i + \delta^*_i$

Update $w \leftarrow w + \delta^*_i y_i x_i$

Different mechanisms on accessing $w$ in the shared memory.


Huan and Hsieh, “Fixing the Convergence Problems in Parallel Asynchronous Dual Coordinate Descent”, in ICDM 2016
Other multi-core algorithms for SVM

- Asynchronous stochastic (sub-)gradient decent for the primal problem:

- Another approach (parallelized variable selection)
  Chiang et al., “Parallel Dual Coordinate Descent Method for Large-scale Linear Classification in Multi-core Environment”, in KDD 2016.

- Other parallel primal solvers:
Large-scale linear ERM: out-of-core version

- **Question:** how to solve linear SVM on a single machine when *data cannot fit in memory*?
- **Dual coordinate descent:** need random access, not suitable when training samples are stored in disk
Large-scale linear: out-of-core version

- **Block coordinate descent:**
  - Partition data into blocks $S_1, \ldots, S_k$
  - For $t = 1, 2, \ldots$
    - Load a block $S_i$ into memory
    - Update this block of dual variables by dual coordinate descent

  (Yu et al., “Large Linear Classification when Data Cannot Fit In Memory”, in KDD 2010)

- **Selective block coordinate descent:** keep important samples in memory
  (Chang and Roth, “Selective Block Minimization for Faster Convergence of Limited Memory Large-scale Linear Models”, in KDD 2011)

- **StreamSVM:** one thread keep loading data, while another thread keep running coordinate descent updates
  (Matsushima et al., “Linear Support Vector Machines via Dual Cached Loops”, in KDD 2012)
Comparisons

On webspam dataset, 0.35 million samples, 16.61 million features, dataset size 20.03GB
Distributed Linear ERM

- Each machine stores a subset of training samples
- Each machine conducts dual coordinate updates and has a local $w$
- How to communicate and synchronize the updates?
- Can the methods generalize to other local solvers?
- What if there are millions of machines (nodes), and data is non-iid distributed?
Distributed Linear ERM

- COCOA: local dual coordinate descent updates, and averaging.
  (Jaggi et al., “Communication-Efficient Distributed Dual Coordinate Ascent”, in NIPS, 2014.)

- Block Quadratic Optimization (COCOA with improved step size selection).
  (Lee et al., “Distributed Box-Constrained Quadratic Optimization for Dual Linear Support Vector Machines”, in ICML, 2015.)

- COCOA+: improved COCOA by solving a modified subproblem.
  (Ma et al., “Adding vs. Averaging in Distributed Primal-Dual Optimization”, in ICML, 2015.)

- A more general framework (for other regularizations).
Other distributed algorithms

- **Primal solver:**
  

- **DANE:** distributed Newton-type method
  
  (Shamir et al., “Communication-Efficient Distributed Optimization using an Approximate Newton-type Method”, in ICML 2014.)

- **DisCO:** based on inexact Newton method:
  
Questions?