Part 2: Structured Learning

Kai-Wei Chang

Microsoft Research → Univ. of Virginia
Part 2: Learning a Structured Prediction Model (45min)

• Definition of structured learning
• Local Learning v.s. Global Learning
• Global Learning Algorithms
  • Online learning: Structured Perceptron
  • Batch learning: Structured SVM
• Optimization methods for Structured SVM
  • Stochastic Gradient Decent
  • Dual Coordinate Descent
  • Learning on a multi-core machine
CCM Formulations

\[ y = \arg\max_{y \in Y} \mathbf{w}^T \phi(x, y) + \mathbf{u}^T \mathbf{C}(x, y) \]

This part of the tutorial focuses on learning \( \mathbf{w} \) (and \( \mathbf{u} \))
Learning a Structured Prediction Model

- **Input:** $x \in X$
- **Truth:** $y^* \in Y(x)$
- **Predicted:** $h(x) \in Y(x)$
- **Loss:** $\Delta(y, y^*)$

**Goal:** find $h \in H$ such that $h(x) \in Y(X)$ minimizing $E_{(x,y) \sim D} \left[ \Delta(y, h(x)) \right]$ based on $N$ samples $(x_n, y_n) \sim D$
Learning Paradigms [Punyakanok+ 05]

- Local Learning
  - Learning local models independently
  - Ensure output is coherent at test time
  - E.g., One-against-all multiclass classification

- Global Learning
  - Learning with inference
  - Training and testing are consistent
  - Constrained classification for multiclass [Har-Peled et. el 2002; Crammer et. al 2002]
Visualizing One-vs-all

From the full dataset, construct three binary classifiers, one for each class:

\[ w_{\text{blue}}^T x > 0 \] for blue inputs

\[ w_{\text{red}}^T x > 0 \] for red inputs

\[ w_{\text{green}}^T x > 0 \] for green inputs

Winner Take All will predict the right answer. Only the correct label will have a positive score.

Notation: Score for blue label
One-vs-all may not always work

Black points are not separable with a single binary classifier

*The decomposition will not work for these cases!*

\[ \mathbf{w}_{\text{blue}} \mathbf{x} > 0 \quad \text{for blue inputs} \]
\[ \mathbf{w}_{\text{red}} \mathbf{x} > 0 \quad \text{for red inputs} \]
\[ \mathbf{w}_{\text{green}} \mathbf{x} > 0 \quad \text{for green inputs} \]

???
Local Learning: One-vs-all classification

- Easy to learn
  - Use any binary classifier learning algorithm

Problems

- Calibration issues
  - We are comparing scores produced by K classifiers trained independently. No reason for the scores to be in the same numerical range!

- Might not always work
  - Yet, works fairly well in many cases, especially if the underlying binary classifiers are tuned, regularized
Global Learning Motivation

- Decomposition methods
  - Do not account for how the final predictor will be used
  - Do not optimize any global measure of correctness

- **Goal:** To train a multiclass classifier that is “global”
Global “One-vs-all” Approach

- **Idea:** Create \( K \) classifiers \( \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K \). For examples with label \( i \), we want
  \[
  \mathbf{w}_i^T \mathbf{x} > \mathbf{w}_j^T \mathbf{x} \text{ for all } j
  \]

- **Prediction:** \( \arg \max_i \mathbf{w}_i^T \mathbf{x} \)

- **Training:** For each training example \((\mathbf{x}_i, y_i)\) :
  \[
  \hat{y} \leftarrow \arg \max_j \mathbf{w}_j^T \phi(\mathbf{x}_i, y_i)
  \]
  
  if \( \hat{y} \neq y_i \)

  \[
  \mathbf{w}_{y_i} \leftarrow \mathbf{w}_{y_i} + \eta \mathbf{x}_i \quad \text{(promote)}
  \]

  \[
  \mathbf{w}_{\hat{y}} \leftarrow \mathbf{w}_{\hat{y}} - \eta \mathbf{x}_i \quad \text{(demote)}
  \]

\( \eta \): learning rate
Joint Inference with General Constraint Structure
[Roth&Yih’04,07,...]

Recognizing Entities and Relations

Bernie’s wife, Jane, is a native of Brooklyn

Models could be learned separately/jointly; constraints may come up only at decision time.
Training Methods

**Learning + Inference (L+I)**
Learn models independently

**Inference Based Training (IBT)**
Learn one model, all y’s together!

**Intuition:** Learning with constraints may make learning more difficult
Local Learning v.s. Global Learning

❖ Advantages of local training:
  ❖ Modular, very simple to implement
  ❖ Often efficient

❖ Advantages of global training:
  ❖ Models directly capture correlations between outputs (have better theoretical guarantees)
  ❖ Might achieve better performance when data is sufficient
Structured Prediction: Learning

- Learning is thus driven by the attempt to find a weight vector $w$ such that for each given annotated example $(x_i, y_i)$:

\[
\text{Score of annotated structure} \geq \text{Score of any other structure} + \text{Penalty for predicting other structure}
\]

- The update of the weight vector $w$ can be done in an on-line or a batch fashion
Global Learning Algorithms

- **Online** e.g., Structured Perceptron [Collins 02]
  - Receive an instance; and update

- **Batch** e.g., Structured SVM [Taskar+05]
  - Collect a set of data; formulate learning as an optimization problem

Solve inferences  Update the model
Structured Perceptron

- **Perceptron** (binary classification):
  
  For each training example \((x_i, y_i)\):
  
  \[
  \hat{y} \leftarrow \text{sgn}(w^T x_i)
  \]
  
  if \(y_i \neq \hat{y}\), then \(w \leftarrow w + \eta y_i x_i\)

- **Structured Perceptron**

  For each training example \((x_i, y_i)\):
  
  \[
  \hat{y} \leftarrow \arg \max_{y \in Y} w^T \phi(x_i, y_i)
  \]
  
  \[
  w \leftarrow w + \eta' [\phi(x_i, y_i) - \phi(x_i, \hat{y})]
  \]

Perceptron is a special case when

\[
\phi(x, y) = yx, \quad \eta' = \frac{1}{2} \eta
\]
Structured Perceptron Variations

- *(Marginal) Structured Perceptron*
  
  For each training example \((x_i, y_i)\):
  
  \[
  \hat{y} \leftarrow \arg \max_{y \in Y} w^T \phi(x_i, y_i) + \Delta(y, y_i)
  \]
  
  \[
  w \leftarrow w + \eta' [\phi(x_i, y_i) - \phi(x_i, \hat{y})]
  \]

- Parallel Structured Perceptron [McDonald et al 10]:
  
  1. Split data into \(p\) parts.
  2. Train Structured Perceptron on each data block in parallel.
  3. Mixed the models using a linear combination.
  4. Repeat Step 2 and use the mixed model as the initial model.
Recap

- Learning is thus driven by the attempt to find a weight vector $w$ such that for each given annotated example $(x_i, y_i)$:

- Introduce slack variables $\{\xi_i\}$

$$w^T \phi(x_i, y_i) - w^T \phi(x_i, y) \geq \Delta(y_i, y) - \xi_i$$
Structured SVM (Batch)
(Tsochantaridis et al. 05)

Given a set of training examples \( D = \{x_i, y_i\}_{i=1}^l \)
\[
\min_{w,\xi} \frac{1}{2} w^T w + C \sum_i \xi_i \\
\text{s.t. } w^T \Phi(x_i, y_i) - w^T \Phi(x_i, y) \geq \Delta(y_i, y) - \xi_i \\
\forall i, y \in Y_i
\]

Note:
1. \( w^T \Phi(x, y) \) is the scoring function used in inference.
2. Equivalent to the following optimization problem:
\[
\min_w \frac{1}{2} w^T w + C \sum_i \left( \max_y [\Delta(y_i, y) + w^T \Phi(x_i, y)] - w^T \Phi(x_i, y_i) \right)
\]
Optimization Methods for Structured SVM

- **Online** e.g., Structured Perceptron [Collins 02]
- **Batch** e.g., Structured SVM [Taskar+05]
  - Cutting plane: [Tsochantaridis+ 05, Joachims+ 09]
  - Dual Coordinate Descent: [Shevade+ 11, Chang+ 13]
  - Block-Coordinate Frank-Wolfe: [Lacoste-Julien+ 13]
  - Parallel Dual Coordinate Descent: [Chang+ 13a]

Solve inferences  Update the model
Stochastic (sub-)gradient descent

To minimize a function $g(z)$ that has the form $\sum_i g_i(z)$

- Initialize $z_0$
- Iterate until convergence
  - Pick a random $g_i$ and compute its (sub)gradient at $z_t$: $\nabla g_i(z_t)$
  - Update: $z_{t+1} \leftarrow z_t - \gamma_t \nabla g_i(z_t)$ \hspace{1cm} $\gamma_t$: learning rate

General idea: Replace the gradient with a noisy estimate
Sub-gradient computation

\[ \min_w \frac{1}{2} w^T w + C \sum_i \left( \max_y [\Delta(y_i, y) + w^T \phi(x_i, y)] - w^T \phi(x_i, y_i) \right) \]

- Solve the max. Suppose solution is \( y' \)
- The loss-augmented/loss-sensitive/cost-augmented inference step

- Compute gradient of

\[ \frac{1}{2} w^T w + C \left( [\Delta(y_i, y) + w^T \phi(x_i, y)] - w^T \phi(x_i, y_i) \right) \]

- Subgradient is

\[ w + C \left( \phi(x_i, y') - \phi(x_i, y_i) \right) \]
SGD for structural SVM: The update

- Solve inference and compute sub-gradient:
  \[ w + C \left( \phi(x_i, y') - \phi(x_i, y_i) \right) \]

- At each step, go down the gradient:
  \[ w \leftarrow w - \gamma_t \left( w + C \left( \phi(x_i, y') - \phi(x_i, y_i) \right) \right) \]

- Compared to Structured Perception update:
  \[ w \leftarrow w + \eta' \left[ \phi(x_i, y_i) - \phi(x_i, \hat{y}) \right] \]
  \[ (\gamma_t, \eta: \text{learning rate}) \]
L2-loss Structured SVM

Given a set of training examples $D = \{x_i, y_i\}_{i=1}^l$

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

$$\text{s.t. } w^T \phi(x_i, y_i) - w^T \phi(x_i, y) \geq \Delta(y_i, y) - \xi_i \quad \forall i, y \in Y_i$$

- Solve problem in the dual space
  - Recall that in dual SVM: one dual variable for each instance
  - In structured case: one for (instance, output assignment)
Dual Problem of Structural SVM

\[
\min_{\alpha > 0} D(\alpha), \quad \text{and} \\
D(\alpha) = \frac{1}{2} \left\| \sum_{i, y} \alpha_{i, y} \phi(y, y_i, x_i) \right\|^2 + \frac{1}{4C} \sum_i \left( \sum_y \alpha_{i, y} \right)^2 - \sum_{i, y} \Delta(y, y_i) \alpha_{i, y},
\]

where \( \phi(y, y_i, x_i) = \phi(y_i, x_i) - \phi(y, x_i) \).

• Number of \( \alpha \) variables can be exponentially large.
• Relationship between \( w^* \) and \( \alpha^* \)

\[
w^* = \sum_{i, y} \alpha_{i, y}^* \phi(y, y_i, x_i).
\]

For linear model: maintain the relationship between \( w \) and \( \alpha \) throughout the learning process [Hsieh et.al. 08].
Structured Learning by Dual Coordinate Descent

- Number of dual variables can be exponentially large
- Maintain an active set \( A \) of dual variables:
  - Identify dual variables that will be likely non-zero
- Single-thread implementation:
  - Select and maintain \( A \) (active set selection step).

\[
\max_{y \in \mathcal{Y}_i} \ w^T \phi(x_i, y) + \Delta(y_i, y)
\]

- Update the values of \( \alpha_{i,y} \in A \) (learning step).
  - (Approximately) solving a sub-problem.
A parallel Dual Coordinate Descent Algorithm

A Master-Slave architecture (MS-DCD):

- Given $p$ processors: split data into $p$ parts.
- At each iterations:
  - Master sends current model to slave threads.
  - Each slave thread solves loss-augmented inference problems associated with a data block and updates the active set.
  - After all slave threads finish, master thread updates the model $w$ according to the active set.
DEMI-DCD

DEMI-DCD: Decouple Model-update and Inference with Dual Coordinate Descent.

- Let $p$ be #threads, and split training data into $B_1, B_2, \ldots, B_{p-1}$.
- Active set selection thread $j$: select and maintain the active set $A_i$ for each example $i$ in $B_j$.
- Learning thread: loop over all examples and update model $w$.
- $A$ and $w$ are shared between threads using shared memory buffers.
Animation

Buffer

Learning

Inference

$\{\alpha_{i,y}\}$ variables
Convergence on Primal Function Value

Relative primal function value difference along training time (Entity-Relation task)
Test Performance

Test Performance (F1 score) along training time
• **Part 2**: Learning a Structured Prediction Model
  - Definition of structured learning
  - Local Learning v.s. Global Learning
  - Global Learning Algorithms
    - Online learning: Structured Perceptron
    - Batch learning: Structured SVM
  - **Optimization methods for Structured SVM**
    - Stochastic Gradient Decent
    - Dual Coordinate Descent
    - Learning on a multi-core machine

In part 4, we will describe how to use them in practice