Learning Structural SVMs with Latent Variables

Chun-Nam John Yu, Thorsten Joachims

Presenter: Jacob Kahn (jacokahn)

October 19, 2017

Overview

Motivating Problem: Noun Phrase Coreferencing

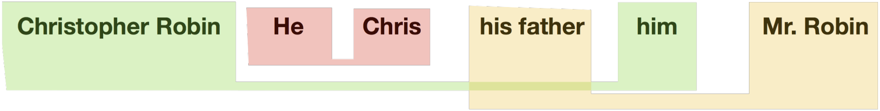
- **Task:** determine which noun phrases in some piece of text refer to the same entity.

Christopher Robin is alive and well. **He** lives in England. **He** is the same person that you read about in the book, Winnie the Pooh. As a boy, **Chris** lived in a pretty home called Cotchfield Farm. When Chris was three years old, **his father** wrote a poem about **him**. The poem was printed in a magazine for others to read. **Mr. Robin** then wrote a book.

- **Correlation clustering:** objective function maximizes the sum of pairwise similarities.

Motivating Problem: Noun Phrase Coreferencing

Christopher Robin is alive and well. **He** lives in England. **He** is the same person that you read about in the book, Winnie the Pooh. As a boy, **Chris** lived in a pretty home called Cotchfield Farm. When Chris was three years old, **his father** wrote a poem about **him**. The poem was printed in a magazine for others to read. **Mr. Robin** then wrote a book.



- For a cluster of size k, there are $O(k^2)$ links, the vast majority of which contain very weak signals.
- Difficult to determine transitive coreference without searching through an entire piece of text.

Motivating Problem: Noun Phrase Coreferencing

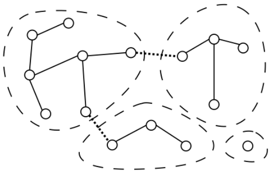


Figure 1. The circles are the clusters defined by the label y. The set of solid edges is one spanning forest h that is consistent with y. The dotted edges are examples of incorrect links that will be penalized by the loss function.

- Here, $\mathcal{Y}$ is the set of non-contradictory pairwise clusters.
- Instead, model as an agglomeration problem.
  - **Input:** x, contains n noun phrases, and pairwise features $\boldsymbol{x}_{ij}$ between the ith and jth noun phrases.
  - **Output:** y, which is a partition of the N phrases into coreferent clusters.
  - To choose which clusters are strong, put a **latent variable** h, which is a spanning forest of *strong* coreference links that is consistent with y.

Structured SVM (SSVM)

Given examples $\mathcal{D} = \{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{l}$. Say $\boldsymbol{x}_i \in \mathcal{X}$. The following applies margin rescaling (Tsochantaridis et al., 2004) to give a smooth, convex upper bound.

Optimization Problem

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

$$\text{such that for } 1 \leq i \leq n, \forall \hat{y} \in \mathcal{Y},$$

$$\boldsymbol{w}^T\Phi(\boldsymbol{x}_i, \boldsymbol{y}_i) - \boldsymbol{w}^T\Phi(\boldsymbol{x}_i, \hat{\boldsymbol{y}}) \geq \Delta(\boldsymbol{y_i}, \hat{\boldsymbol{y}}) - \xi_i$$

- $\Phi(\boldsymbol{x}, \boldsymbol{y})$: feature vector from input $\boldsymbol{x}$ and output $\boldsymbol{y}$
- ξ : loss to minimize
- $\xi_i \geq 0$: slack, penalizes violation
- $\Delta(\boldsymbol{y}_i, \hat{\boldsymbol{y}})$: controls margin between incorrect predictions $\hat{y}$ and correct label

Extending the Structured SVM to Latent Variables

Sometimes, $(x, y) \in \mathcal{X} \times \mathcal{Y}$ is not sufficient to characterize the input-output relationship, but also may depend on a set of latent variables (typically unobserved).

How do we enable the structural SVM to handle latent variables?

Notation: let h be a particular variable in a set of latent variables $\mathcal{H}$. h describes some structure-determining, unobserved factor.

Things to consider:

- Feature representation, loss function
- Training objective that is non-convex
- Inference techniques and problems

Prediction Rules for a Latent Structural SVM

- Extend the joint feature map $\Phi(x, y)$ to $\Phi(x, y, h)$. The feature vector now captures a relation between some input, some output, and some latent variable.
- We now must perform *joint inference* over y and h, and we can mutate the prediction rule for some $f_{\boldsymbol{w}}(x)$ as follows:

New Argmax Prediction Rule

$$f_{\boldsymbol{w}}(x) = (\bar{y}, \bar{h}) = \text{argmax}_{(y,h) \in \mathcal{Y} \times \mathcal{H}} [\boldsymbol{w} \cdot \Phi(x, y, h)]$$

Latent Structural SVM Formulation

Optimization Problem for Latent Structural SVM

$$\min_{\boldsymbol{w},\xi} \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i=1}^{n}\xi_i$$

such that for $1 \leq i, \forall \hat{y} \in \mathcal{Y}$,

$$\max_{h\in\mathcal{H}}[\boldsymbol{w}\cdot\Phi(x_i, y_i, h)] - \max_{\hat{h}\in\mathcal{H}}[\boldsymbol{w}\cdot\Phi(x_i, \hat{y}, \hat{h})] \leq \Delta(y_i, \hat{y}, \hat{h}) - \xi_i$$

$\Phi(\boldsymbol{x}, \boldsymbol{y}, h)$: feature vector from input $\boldsymbol{x}$, output $\boldsymbol{y}$, and latent variable h
$\Delta(\boldsymbol{y}_i, \boldsymbol{y}, h)$: margin; assumes no dependence on latent h
$\xi_i \geq 0$: slack, penalizes violation, which now upper bounds the loss

- If the latent variable is not present, the model degenerates to a structural SVM

Prediction Loss with the Addition of Latent Variables

Bound on constraint loss in structural SVM (without latent variable)

$$\Delta(y_i, f_{\mathbf{w}}(x_i)) \leq \overbrace{\max_{\hat{y}\in\mathcal{Y}} [\mathbf{w}\cdot\Phi(x_i,\hat{y}) + \Delta(y_i,\hat{y})]}^{\text{convex}} - \underbrace{\mathbf{w}\cdot\Phi(x_i,y_i)}_{\text{linear}} = \xi_i$$

We now need to take the maximum over all latent variables h in $\mathcal{H}$.

Bound on constraint loss in latent structural SVM

$$\Delta(y_i, f_{\mathbf{w}}(x_i)) \leq \underbrace{\max_{(\hat{y},\hat{h})\in\mathcal{Y}\times\mathcal{H}} [\mathbf{w}\cdot\Phi(x_i,\hat{y},\hat{h}) + \Delta(y_i,\hat{y},\hat{h})]}_{\text{convex}} - \underbrace{\max_{h\in\mathcal{H}}[\mathbf{w}\cdot\Phi(x_i,y_i,h)]}_{\text{concave}} = \xi_i$$

Latent Structural SVM Objective Formulation

Attempting to formulate the problem in the dual, a concave constraint remains, as we must compute the maximum over $\mathcal{H}$:

Objective function, with latent variable, dual formulation

$$\min_{\boldsymbol{w}} \overbrace{\left[\frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i=1}^{n} \max_{(\hat{y},\hat{h})\in\mathcal{Y}\times\mathcal{H}} [\boldsymbol{w}\cdot\Phi(x_i,\hat{y},\hat{h}) + \Delta(y_i,\hat{y},\hat{h})]\right]}^{\text{convex}} - \underbrace{\left[C\sum_{i=1}^{n}\max_{h\in\mathcal{H}}[\boldsymbol{w}\cdot\Phi(x_i,y_i,h)]\right]}_{\text{concave}}$$

The CCCP Algorithm for Non-Convex Objectives

- We have a term with convex and concave parts. How to proceed?
- Concave-Convex optimization procedure (Yuille and Rangarajan '03)

Algorithm:

1. Decompose the objective into a convex and concave part.
2. Upper bound the concave part with a hyperplane.
3. Minimize the resulting convex sum.
4. Iterate on the above until convergence.

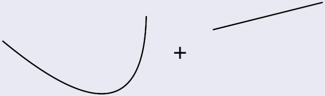
The CCCP Algorithm for Non-Convex Objectives

The Concave-Convex Algorithm:

1. Decompose objective into convex and concave part:

2. Upper bound concave part with a hyperplane:

3. Minimize resulting convex sum (iterate until convergence is reached):



Applying CCCP to the Objective

We can think of computing the upper bounding hyperplane in the CCCP algorithm as finding the latent variable that **best explains the input-output pair** (x_i, y_i). This is equivalent to **computing the upper bounding hyperplane** on the concave problem of selecting the best $h \in \mathcal{H}$.

Let h_i^* be that best chosen latent variable from $\mathcal{H}$, equivalently defined as:

"Completing" the latent variables

$$h_i^* = \text{argmax}_{h \in \mathcal{H}} \boldsymbol{w} \cdot \Phi(x_i, y_i, h)$$

Applying CCCP to the Objective

Now, we've converted the concave latent variable selection problem into a linear term, and we have a final, convex objective:

Latent structural SVM objective with upper bounding hyperplane

$$\min_{\boldsymbol{w}} \overbrace{\left[\frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i=1}^{n} \max_{(\hat{y},\hat{h})\in\mathcal{Y}\times\mathcal{H}} [\boldsymbol{w}\cdot\Phi(x_i,\hat{y},\hat{h}) + \Delta(y_i,\hat{y},\hat{h})]\right]}^{\text{convex}} - \underbrace{\left[C\sum_{i=1}^{n} \boldsymbol{w}\cdot\Phi(x_i,y_i,h_i^*)\right]}_{\text{linear}}$$

From here, we can apply cutting plane algorithms like we can apply to any structural SVM.

Latent Structural SVM Summary

Final Optimization Problem

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

such that for $1 \leq i, \forall \hat{y} \in \mathcal{Y}$,

$$\max_{h\in\mathcal{H}}[\mathbf{w}\cdot\Phi(x_i, y_i, h)] - \max_{\hat{h}\in\mathcal{H}}[\mathbf{w}\cdot\Phi(x_i, \hat{y}, \hat{h})] \leq \Delta(y_i, \hat{y}, \hat{h}) - \xi_i$$

Three primary inference problems overall:

Prediction : $\text{argmax}_{(y,h)\in\mathcal{Y}\times\mathcal{H}}\mathbf{w}\cdot\Phi(x_i, y, h)$

Loss-augmentation : $\text{argmax}_{(\hat{y},\hat{h})\in\mathcal{Y}\times\mathcal{H}}[\mathbf{w}\cdot\Phi(x_i, \hat{y}, \hat{h} + \Delta(y_i, \hat{y}, \hat{h})]$

Latent var. determination : $\text{argmax}_{h\in\mathcal{H}}\mathbf{w}\cdot\Phi(x_i, y_i, h)$

Noun Phrase Coreferencing with Clustering

We can determine a clustering y given an input x with an maximum spanning tree algorithm (Kruskal's algorithm), where weights for an edge (i, j) can be written as $\boldsymbol{w} \cdot \boldsymbol{x}_{ij}$.

Clustering score with latent spanning forest

$$\boldsymbol{w} \cdot \Phi(x, y, h) = \sum_{(i,j) \in h} \boldsymbol{w} \cdot \boldsymbol{x}_{ij}$$

- Only consider edges (i, j) that are in the latent spanning forest.
- Output the clustering defined by the forest h as y (prediction).

Noun Phrase Coreferencing with Clustering - Loss

Loss function

$$\Delta(y, \hat{y}, \hat{h}) = n(y) - k(y) - \sum_{(i,j)\in h} l(y, (i,j))$$

$n(y)$	: number of vertices in the correct clustering y
$k(y)$	: number of edges in the correct clustering y
$l(y, (i,j))$	: 1 if i and j are same-clustered in y, else -1

Works well, since we can back out $\hat{h}$, and can compute loss-augmented inference with Kruskal's algorithm. We can also use Kruskal's algorithm to complete h (to choose the optimal, in $\mathcal{H}$.

Noun Phrase Coreferencing with Clustering - Results

Table 2. Clustering Accuracy on MUC6 Data

	MITRE Loss	Pair Loss
SVM-cluster	41.3	2.89
Latent Structural SVM	44.1	2.66
Latent Structural SVM (modified loss, $r = 0.01$)	**35.6**	4.11

- Start with the spanning forest as a linear chain (chronological order); the algorithm then inserts new weights.
- Modifications to incorrect-cluster-linking penalty were required (significant decreases: mistakes were over-penalized).
- Overall improvement once penalization decreased.

References

Chun-Nam John Yu, Thorsten Joachims
Learning Structural SVMs with Latent Variables
ICML, 2009

Tsochantaridis et al.
Support Vector Machine Learning for Interdependent and Structured Output Spaces
ICML, 2004

A. L. Yuille and Anand Rangarajan
The Concave-Convex Procedure (CCCP)
NIPS, 2002

Kai-Wei Chang, Vivek Srikumar, and Dan Roth
Multi-core Structural SVM Training
ECML, 2013

Ming-Wei Chang
Structured Prediction with Indirect Supervision
UIUC, 2011

Questions?