Learning Structural SVMs with Latent Variables

Chun-Nam John Yu CNYU@CS.CORNELL.EDU
Thorsten Joachims TJ@CS.CORNELL.EDU
Department of Computer Science, Cornell University, Ithaca NY 14853 USA

1. Introduction

It is well known in statistics and machine learning that the combination of latent (or hidden) variables and observed variables offer more expressive power than models with observed variables alone. Latent variables can be used to model explanatory factors that cannot be measured in experiments, or to control the number of degrees of freedom of a model that generates the observed data. Well-known classic examples include mixture models, factor analysis, K-means, and PCA.

In structured output prediction, there have been various applications of latent variable models. For example, it has been used for capturing interesting substructures or parts in object recognition [10], for automatic refinement of grammars in parsing [6], and for dimensionality reduction in people tracking [4]. Almost all of these latent variable models are probabilistic in nature and use EM or gradient-based methods to optimize the non-convex objective in training.

The use of latent variables is less well-explored in the case of large-margin structured output learning such as Max-Margin Markov Networks or Structural SVMs [7, 8]. These models are non-probabilistic and offer excellent performance in many structured prediction tasks in the fully observed case. Currently, they do not support the use of latent variables, which excludes many interesting applications. In this work we extend Structural SVMs to include latent variables, and provide an efficient algorithm for solving the optimization problem of our proposed formulation. We apply our new algorithm to the problem of discriminative motif finding in yeast DNA and some initial results are presented.

2. Structural SVMs

Suppose we are given a training set of input-output structure pairs $\mathcal{S}=\{(x_1,y_1),\dots,(x_n,y_n)\}\in(\mathcal{X}\times\mathcal{Y})^n$. We want to learn a linear prediction rule of the form

$$f_{\vec{w}}(x)=\text{argmax}_{y\in\mathcal{Y}}\,\vec{w}\cdot\Phi(x,y), \qquad (1)$$

where Φ is a joint feature vector that describes the relationship between input x and structured output y, with $\vec{w}$ being the parameter vector.

In structured learning we are usually given a loss function Δ to measure how much the predicted structured output $f_{\vec{w}}(x)$ differs from the correct output. We want to learn a prediction rule that incurs small average loss on future inputs.

According to the Empirical Risk Minimization (ERM) principle [9], we should search for a parameter vector $\vec{w}$ with low empirical risk $\sum_{i=1}^{n}\Delta(y_i,f_{\vec{w}}(x_i))$. But in general this is very difficult due to non-convexity and discontinuity of the loss function Δ and the exponentially many possible structures $f_{\vec{w}}(x_i)$ in the output space $\mathcal{Y}$.

The Structural SVM formulation [8] overcomes these difficult issues by replacing the loss function Δ with a piecewise linear convex upper bound (margin rescaling)

$$\Delta(y_i,\hat{y}_i(\vec{w}))\le\max_{\hat{y}\in\mathcal{Y}}[\Delta(y_i,\hat{y})+\vec{w}\cdot\Phi(x_i,\hat{y})]-\vec{w}\cdot\Phi(x_i,y_i)$$

where $\hat{y}_i(\vec{w})=\text{argmax}_{y\in\mathcal{Y}}\,\vec{w}\cdot\Phi(x_i,y)$.

Instead of minimizing the true empirical risk, structural SVMs solve an easier convex optimization problem:

$$\min_{\vec{w}}\frac{1}{2}\|\vec{w}\|^2+C\sum_{i=1}^{n}\left[\max_{\hat{y}\in\mathcal{Y}}[\Delta(y_i,\hat{y})+\vec{w}\cdot\Phi(x_i,\hat{y})]-\vec{w}\cdot\Phi(x_i,y_i)\right]$$

Minimizing this convex upper bound gives excellent performance on many structured prediction tasks.

3. Structural SVMs with Latent Variables

Generalizing structural SVMs, we are interested in structured prediction problems where the input-output relationship is not completely characterized by the $(x,y)\in\mathcal{X}\times\mathcal{Y}$ pairs in the training set alone, but also depends on a set of unobserved latent variables $h \in \mathcal{H}$. To generalize the structural SVM formulation, we extend our joint feature vector $\Phi(x, y)$ to accept three arguments $\Phi(x, y, h)$ describing the relation among input x, output y, and latent variables h. We want to learn a prediction rule of the form

$$f_{\vec{w}}(x) = \bar{y}$$

where $(\bar{y}, \bar{h}) = \mathrm{argmax}_{(y,h)\in\mathcal{Y}\times\mathcal{H}}\, \vec{w}\cdot\Phi(x, y, h)$.

Ideally according to the ERM principle we would like to minimize the empirical risk

$$\sum_{i=1}^{n} \Delta((y_i, h_i^*(\vec{w})), (\hat{y}_i(\vec{w}), \hat{h}_i(\vec{w})))$$

where $h_i^*(\vec{w}) = \mathrm{argmax}_{h\in\mathcal{H}}\, \vec{w}\cdot\Phi(x_i, y_i, h)$ and $(\hat{y}_i(\vec{w}), \hat{h}_i(\vec{w})) = \mathrm{argmax}_{(y,h)\in\mathcal{Y}\times\mathcal{H}}\, \vec{w}\cdot\Phi(x_i, y, h)$.

Note that the loss function Δ is extended to take into account the latent variables $h \in \mathcal{H}$. Essentially we want to minimize the loss between the pair $(\hat{y}_i(\vec{w}), \hat{h}_i(\vec{w}))$ given by the prediction rule and the best latent variable $h_i^*(\vec{w})$ that explains the input-output pair (x_i, y_i) in the training set.

As already discussed above, minimizing the empirical risk directly is in general very difficult and we seek an upper bound on the loss:

$$\begin{aligned}
&\Delta((y_i, h_i^*(\vec{w})), (\hat{y}_i(\vec{w}), \hat{h}_i(\vec{w}))) \\
\leq\ &\Delta((y_i, h_i^*(\vec{w})), (\hat{y}_i(\vec{w}), \hat{h}_i(\vec{w}))) \\
&- [\vec{w}\cdot\Phi(x_i, y_i, h_i^*(\vec{w})) - \vec{w}\cdot\Phi(x_i, \hat{y}_i(\vec{w}), \hat{h}_i(\vec{w}))] \\
=\ &\vec{w}\cdot\Phi(x_i, \hat{y}_i(\vec{w}), \hat{h}_i(\vec{w})) + \Delta((y_i, h_i^*(\vec{w})), (\hat{y}_i(\vec{w}), \hat{h}_i(\vec{w}))) \\
&- \vec{w}\cdot\Phi(x_i, y_i, h_i^*(\vec{w})) \\
=\ &\left(\max_{(\hat{y},\hat{h})\in\mathcal{Y}\times\mathcal{H}} \vec{w}\cdot\Phi(x_i, \hat{y}, \hat{h})\right) + \Delta((y_i, h_i^*(\vec{w})), (\hat{y}_i(\vec{w}), \hat{h}_i(\vec{w}))) \\
&- \left(\max_{h\in\mathcal{H}} \vec{w}\cdot\Phi(x_i, y_i, h)\right)
\end{aligned} \tag{2}$$

In the case of structural SVMs without latent variables, the complex dependence on $\vec{w}$ within the loss Δ can be got rid of with the following inequality:

$$\left(\max_{\hat{y}\in\mathcal{Y}} \vec{w}\cdot\Phi(x, \hat{y})\right) + \Delta(y_i, \hat{y}_i(\vec{w})) \leq \max_{\hat{y}\in\mathcal{Y}} [\vec{w}\cdot\Phi(x_i, \hat{y}_i) + \Delta(y_i, \hat{y})] \tag{3}$$

The right hand side of (3) is commonly referred to as loss-augmented inference in structural SVM training. However in the case with latent variables the dependence of Δ on the latent variables $h_i^*(\vec{w})$ of the correct label y_i prevents us from using loss-augmented inference to remove the dependence on $\vec{w}$ within the loss in (2).

To circumvent this difficulty, we assume that the loss function Δ does not depend on the latent variables:

$$\Delta((y_i, h_i^*(\vec{w})), (\hat{y}_i(\vec{w}), \hat{h}_i(\vec{w}))) = \Delta(y_i, \hat{y}_i(\vec{w}))$$

The bound in (2) then becomes

$$\begin{aligned}
&\Delta((y_i, h_i^*(\vec{w})), (\hat{y}_i(\vec{w}), \hat{h}_i(\vec{w}))) \\
\leq &\left(\max_{(\hat{y},\hat{h})\in\mathcal{Y}\times\mathcal{H}} [\vec{w}\cdot\Phi(x_i, \hat{y}, \hat{h}) + \Delta(y_i, \hat{y})]\right) - \left(\max_{h\in\mathcal{H}} \vec{w}\cdot\Phi(x_i, y_i, h)\right).
\end{aligned}$$

This gives rise to the following optimization problem for structural SVMs with latent variables:

$$\begin{aligned}
\min_{\vec{w}} \frac{1}{2}\|\vec{w}\|^2 &+ C\sum_{i=1}^{n}\left(\max_{(\hat{y},\hat{h})\in\mathcal{Y}\times\mathcal{H}} [\vec{w}\cdot\Phi(x_i, \hat{y}, \hat{h}) + \Delta(y_i, \hat{y})]\right) \\
&- C\sum_{i=1}^{n}\left(\max_{h\in\mathcal{H}} \vec{w}\cdot\Phi(x_i, y_i, h)\right)
\end{aligned} \tag{4}$$

It is easy to observe that the above formulation reduces to the usual structural SVM formulation in the absence of latent variables.

The assumption on the loss Δ is important because:

(i) it allows us to decompose the bound on the loss into the sum of a convex and concave part, allowing efficient algorithms such as CCCP to be used for its solution

(ii) the assumption is naturally satisfied by many structured prediction tasks with latent variables. In many real world applications such as parsing and object recognition mentioned in the introduction, the latent variables serve as indicator for mixture components or intermediate representations and are not part of the output. The loss that we are interested in for these tasks do not depend on the latent variables.

(iii) it distinguishes our approach from transductive structured output learning [12]. When the loss function Δ depends only on the fully observed label y_i, it rules out the possibility of transductive learning, but the restriction also results in simpler optimization problems compared to the transductive cases (for example, the approach in [12] involves constraint removals to deal with dependence on $h_i^*(\vec{w})$ within the loss Δ).

4. Solving the Optimization Problem

The objective of the optimization problem (4) from the last section can be written as the difference of two convex functions:

$$\min_{\vec{w}} \left[\frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{n} \max_{(\hat{y},\hat{h})\in\mathcal{Y}\times\mathcal{H}} [\vec{w}\cdot\Phi(x_i,\hat{y},\hat{h}) + \Delta(y_i,\hat{y})]\right] - \left[C\sum_{i=1}^{n} \max_{h\in\mathcal{H}} \vec{w}\cdot\Phi(x_i,y_i,h)\right]$$

This allows us to solve the optimization problem using the Constrained Concave-Convex Procedure (CCCP) [11]. The general template for a CCCP algorithm for minimizing a function $f(\vec{w}) - g(\vec{w})$, where f and g are convex, works as follows:

Algorithm 1 Constrained Concave-Convex Procedure (CCCP)

```
1: Set $t = 0$ and initialize $\vec{w}_0$
2: **repeat**
3:     Find hyperplane $\vec{v}_t$ such that $-g(\vec{w}) \leq -g(\vec{w}_t) + (\vec{w} - \vec{w}_t)\cdot\vec{v}_t$ for all $\vec{w}$
4:     Solve $\vec{w}_{t+1} = \mathrm{argmin}_{\vec{w}}\, f(\vec{w}) + \vec{w}\cdot\vec{v}_t$
5:     Set $t = t + 1$
6: **until** decrease of objective $[f(\vec{w}_t) - g(\vec{w}_t)] - [f(\vec{w}_{t-1}) - g(\vec{w}_{t-1})] < \epsilon$
```

The CCCP algorithm is guaranteed to decrease the objective function at every iteration and to converge to a local minimum [11]. Line 3 constructs a hyperplane that upper bounds the concave part of the objective $-g$, so that the optimization problem solved at line 4 is convex.

In terms of the optimization problem for structural SVMs with latent variables, the step of computing the upper bound for the concave part in line 3 involves computing

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

for each i, and the hyperplane constructed is $\vec{v}_t = \sum_{i=1}^{n}\Phi(x_i,y_i,h_i^*)$.

Computing the new iterate $\vec{w}_{t+1}$ in line 1 involves solving the standard structural SVM optimization problem by completing y_i with the latent variables h_i^* as if they were completely observed:

$$\min_{\vec{w}} \frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{n} \max_{(\hat{y},\hat{h})\in\mathcal{Y}\times\mathcal{H}} [\vec{w}\cdot\Phi(x_i,\hat{y},\hat{h}) + \Delta(y_i,\hat{y})] - C\sum_{i=1}^{n}\vec{w}\cdot\Phi(x_i,y_i,h_i^*)$$

Thus the CCCP algorithm applied to structural SVM with latent variables gives rise to a very intuitive algorithm that alternates between inputing the latent variables h_i^* that best explain the training pair (x_i, y_i) and solving the structural SVM optimization problem while treating the latent variables as completely observed. This is similar to the iterative process of Expectation Maximization (EM) [2]. But unlike EM which maximizes the expected log likelihood under the marginal distribution of the latent variables, we are minimizing the loss against a single latent variable h_i^* that best explains (x_i, y_i).

In our implementation, we used the cutting plane algorithm to solve the standard structural SVM problem of line 1 in Algorithm 1 to a fixed precision $C\epsilon$. The outer loop is also stopped when the decrease in objective is less than $C\epsilon$. In practice the algorithm could be sped up by solving the inner convex QP to a lower precision, and re-using cutting planes generated in one iteration for the next. However, in this workshop paper we leave these options as future exploration.

5. Experiments

Our development of the structural SVM with latent variables was motivated by a motif finding problem in yeast DNA through collaboration with computational biologists. Motifs are repeated patterns in DNA sequences that have or are believed to have biological significance. Our dataset consists of ARSs (autonomously replicating sequences) screened in two yeast species S. kluyveri and S. cerevisiae. Our task is to predict whether a particular sequence is functional in S. cerevisiae and to find out the motif responsible. All the native ARSs in S. cerevisiae are labeled as positive. The ones that showed ARS activity in S. kluyveri were then further tested to see whether they consist a functional ARS in S. cerevisiae. If they did they are labeled as positive, otherwise they are labeled as negative. Altogether we have 124 positive examples and 75 negative examples.

Popular methods for motif finding includes methods based on EM [1] and Gibbs-sampling [3]. For this particular yeast dataset we believe a discriminative approach, especially one incorporating large-margin separation, is beneficial because of the close relationship and DNA sequence similarity among the different yeast species in the dataset.

For a motif of length l, the weight vector contains a position-specific weight matrix with $4 \times l$ parameters (for the 4 bases A, C, G, T), and parameters for the background model (we use a Markov model of order

Table 1. Classification Error on Yeast DNA (10CV)

Algorithm	Error rate
Gibbs sampler ($l = 11$)	37.97%
Gibbs sampler ($l = 17$)	35.06%
Latent Variable Structural SVM ($l = 11$)	35.85%
Latent Variable Structural SVM ($l = 17$)	33.12%

3). Because the positions of the motif in the positive sequences are not observed, we model them as hidden variables h in the feature vector. For a negative sequence, its feature vector representation $\Phi(x, y, h)$ consists of background counts only, while for a positive sequence it consists of counts from the position-specific weight matrix at position h to $h + l$, and background counts in all other positions.

For the positive sequences, we randomly initialized the motif position h uniformly over the whole length of the sequence for the CCCP algorithm. We optimized over the zero-one loss Δ for classification and performed a 10-fold cross validation. We trained models using regularization constant C from $\{0.1, 1, 10, 100, 1000\}$ times the size of the training set (197), and each model is re-trained 5 times using 5 different random seeds. We picked models having the best accuracy on the validation fold and report its accuracy on the test fold.

As control we ran a Gibbs sampler [5] on the same dataset. It reports good results on motif lengths $l = 11$ and $l = 17$, which we compare our algorithm against. The Gibbs sampler is given the unfair advantage that it has access to a separate set of about 6400 intergenic sequences for estimating background probabilities, and it is trained using the same cross validation procedure as our algorithm. We can see in Table 1 that our latent variable structural SVM algorithm is showing comparable or even better classification accuracy than the Gibbs sampler using much less data.

Our algorithm typically converges within 20 iterations of the CCCP algorithm. The 5 different random seeds can lead to solutions at different local minima, and this is more evident for larger values of C. We are currently analyzing the motifs and the weight matrix found by our algorithm. We are also working on improvements to the algorithm, such as better starting positions and methods for incorporating the large set of intergenic sequences for weight estimation of the background model.

6. Conclusions and Future Work

We have presented an algorithm for learning Structural SVMs with latent variables and discussed some preliminary results on its application to the problem of motif finding in yeast DNA. We are planning to explore further applications of the algorithm to tasks in natural language processing and vision involving latent variables. On the algorithmic side there are also many interesting questions such as latent variable initialization and the tradeoff between time spent on the inner optimization problem and the number of outer loop iterations required for convergence in the CCCP algorithm, which we are currently investigating.

References

[1] T.L. Bailey and C. Elkan. Unsupervised Learning of Multiple Motifs in Biopolymers Using Expectation Maximization. *Machine Learning*, 21(1):51–80, 1995.

[2] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[3] CE Lawrence, SF Altschul, MS Boguski, JS Liu, AF Neuwald, and JC Wootton. Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science*, 262(5131):208–214, 1993.

[4] Z. Lu, M.A. Carreira-Perpinan, and C. Sminchisescu. People Tracking with the Laplacian Eigenmaps Latent Variable Model, 2007.

[5] P. Ng and U. Keich. GIMSAN: a Gibbs motif finder with significance analysis. *Bioinformatics*, 24(19):2256, 2008.

[6] S. Petrov and D Klein. Discriminative Log-Linear Grammars with Latent Variables. *Advances in Neural Information Processing Systems*, 20, 2007.

[7] B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. *Advances in Neural Information Processing Systems*, 16:51, 2004.

[8] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the twenty-first international conference on Machine learning*. ACM New York, NY, USA, 2004.

[9] V.N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 2000.

[10] S.B. Wang, A. Quattoni, L.P. Morency, D. Demirdjian, and T. Darrell. Hidden Conditional Random Fields for Gesture Recognition. *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, 2, 2006.

[11] AL Yuille and A. Rangarajan. The Concave-Convex Procedure, 2003.

[12] A. Zien, U. Brefeld, and T. Scheffer. Transductive support vector machines for structured variables. In *Proceedings of the 24th international conference on Machine learning*, pages 1183–1190. ACM Press New York, NY, USA, 2007.