Structured Output Prediction with Support Vector Machines

Thorsten Joachims

Cornell University, Ithaca, NY, USA
tj@cs.cornell.edu
http://www.joachims.org

Abstract. This abstract accompanying a presentation at S+SSPR 2006 explores the use of Support Vector Machines (SVMs) for predicting structured objects like trees, equivalence relations, or alignments. It is shown that SVMs can be extended to these problems in a well-founded way, still leading to a convex quadratic training problem and maintaining the ability to use kernels. While the training problem has exponential size, there is a simple algorithm that allows training in polynomial time. The algorithm is implemented in the SVM-Struct software, and it is discussed how the approach can be applied to problems ranging from natural language parsing to supervised clustering.

1 Introduction

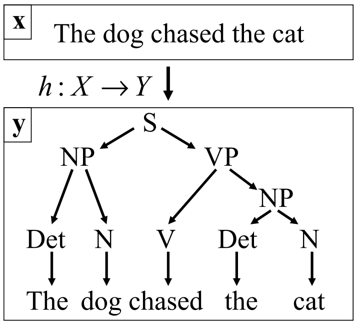
Over the last decade, much of the research on discriminative learning has focused on problems like classification and regression, where the prediction is a single univariate variable. But what if we need to predict complex objects like trees, orderings, or alignments? Such problems arise, for example, when a natural language parser needs to predict the correct parse tree for a given sentence, when one needs to optimize a multivariate performance measure like the F1-score, or when predicting the alignment between two proteins.

This abstract accompanies the presentation at S+SSPR 2006, discussing a support vector approach and algorithm for predicting such complex objects. It summarizes our recent work [10,20,21,11,12,9] on generalizing conventional classification SVMs to a large range of structured outputs and multivariate loss functions, and connects these results to related work [4,3,5,1,16,19,13,23]. While the generalized SVM training problems have exponential size, we show that there is a simple algorithm that allows training in polynomial time. The algorithm is implemented in the SVM-Struct software[1], and it is discussed how the approach can be applied to problems ranging from natural language parsing to supervised clustering.

2 Problems That Require Structured Outputs

While many prediction problems can easily be broken into multiple binary classification problems, other problems require an inherently structured prediction.

[1] Available at `svmlight.joachims.org`



$$\Psi(\mathbf{x},\mathbf{y}) = \begin{array}{c} \left(\begin{array}{c} 1 \\ 0 \\ 2 \\ 1 \\ \vdots \\ 0 \\ 2 \\ 1 \\ 1 \\ 1 \end{array}\right) \end{array} \begin{array}{c} S \to NP\ VP \\ S \to NP \\ NP \to Det\ N \\ VP \to V\ NP \\ \\ Det \to dog \\ Det \to the \\ N \to dog \\ V \to chased \\ N \to cat \end{array}$$

Fig. 1. Illustration of the NLP parsing problem

Consider, for example, the problem of natural language parsing. For a given sentence **x**, the goal is to predict the correct parse tree **y** that reflects the phrase structure of the sentence. This is illustrated on the left-hand side of Figure 1. Training data of sentences that are labeled with the correct tree is available (e.g. from the Penn Tree Bank), making this prediction problem accessible for supervised learning.

Compared to binary classification, the problem of predicting complex and structured outputs differs mainly by the choice of the outputs **y**. What are common structures that we might want to predict?

Trees: We have already discussed the problem of natural language parsing (see e.g. [14]), where a prediction $\mathbf{y} \in \mathcal{Y}$ is a tree.

Sequences: A problem related to parsing is that of part-of-speech tagging (see e.g. [14]). Given a sentence **x** represented as a sequence of words, the task is to predict the correct part-of-speech tag (e.g. "noun" or "determiner") for each word. While this problem could be phrased as a multi-class classification task, it is widely acknowledged that predicting the sequence of tags as a whole allows exploiting dependencies between tags (e.g. it is unlikely to see a verb after a determiner). Similar arguments also apply to tagging protein or gene sequences.

Alignments: For comparative protein structure modelling, it is necessary to predict how the sequence of a new protein with unknown structure aligns against another sequence with know structure (see e.g. [8]). Given the correct alignment, it is possible to predict the structure of the new protein. Therefore, one would like to predict the sequence alignment operations that "best" aligns two sequences according to some cost model.

Equivalence Relation: Noun-phrase co-reference resolution (see e.g. [15]) is the problem of clustering the noun phrases in one documents by whether they refer to the same entity. This can be thought of as predicting an equivalence relation, where training examples are the correct partitionings for some documents. More generally, this problem can be thought of as supervised clustering [9] — training a clustering algorithm to produce the desired kinds of clusters.

While these application problems appear quite different, we will show that they all can be approached in a similar way. In particular, the SVM algorithm we describe in the following is able to address each of these problems.

3 An SVM Algorithm for Structured Outputs

Formally, we consider the problem of learning a function

$$h : \mathcal{X} \longrightarrow \mathcal{Y}$$

where $\mathcal{X}$ is the space of inputs, and $\mathcal{Y}$ is the space of (multivariate and structured) outputs. In the parsing examples, $\mathcal{X}$ is the space of sentences, and $\mathcal{Y}$ is the space of trees over a given set of non-terminal grammar symbols. To learn h, we assume that a training sample of input-output pairs

$$S = ((\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)) \in (\mathcal{X} \times \mathcal{Y})^n$$

is available and drawn i.i.d. from a distribution $P(X, Y)$. The goal is to find a function h from some hypothesis space $\mathcal{H}$ that has low prediction error, or, more generally, low risk

$$\mathcal{R}_P^{\triangle}(h) = \int_{\mathcal{X} \times \mathcal{Y}} \triangle(\mathbf{y}, h(\mathbf{x}))\, dP(\mathbf{x}, \mathbf{y})\,.$$

$\triangle(\mathbf{y}, \hat{\mathbf{y}})$ is a loss function that quantifies the loss associated with predicting $\hat{\mathbf{y}}$ when $\mathbf{y}$ is the correct output value. Furthermore, we assume that $\triangle(\mathbf{y}, \mathbf{y}) = 0$ and $\triangle(\mathbf{y}, \mathbf{y}') \geq 0$ for $\mathbf{y} \neq \mathbf{y}'$. We follow the Empirical Risk Minimization Principle [22] to infer a function h from the training sample S. The learner evaluates the quality of a function $h \in \mathcal{H}$ using the empirical risk $\mathcal{R}_S^{\triangle}(h)$ on the training sample S.

$$\mathcal{R}_S^{\triangle}(h) = \frac{1}{n} \sum_{i=1}^{n} \triangle(\mathbf{y}_i, h(\mathbf{x}_i))$$

Support Vector Machines select an $h \in \mathcal{H}$ that minimizes a regularized Empirical Risk on S. For conventional binary classification where $\mathcal{Y} = \{-1, +1\}$, SVM training is typically formulated as the following convex quadratic optimization problem [6,22].

OP 1 (Classification SVM (primal))

$$\min_{\mathbf{w}, b, \xi_i \geq 0} \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{n}\sum_{i=1}^{n} \xi_i$$
$$s.t.\ \forall i \in \{1, ..., n\}:\ \mathbf{y}_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i$$

To generalize SVM training to structured outputs, we formulate an optimization problem that is similar to multi-class SVMs [7] and generalizes the Perceptron approach described in [4]. The idea is to learn a discriminant function $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ over input/output pairs from which we can derive a prediction by maximizing f over all $\mathbf{y} \in \mathcal{Y}$ for a specific given input $\mathbf{x}$.

$$h_{\mathbf{w}}(\mathbf{x}) = \operatorname*{argmax}_{\mathbf{y}\in\mathcal{Y}} f_w(\mathbf{x}, \mathbf{y})$$

We assume that $f_{\mathbf{w}}(\mathbf{x}, \mathbf{y})$ takes the form of a linear function

$$f_w(\mathbf{x}, \mathbf{y}) = \mathbf{w}^T \Psi(\mathbf{x}, \mathbf{y})$$

where $\mathbf{w} \in \Re^N$ is a parameter vector and $\Psi(\mathbf{x}, \mathbf{y})$ is a feature fector describing the match between input $\mathbf{x}$ and output $\mathbf{y}$. Intuitively, one can think of $f_w(\mathbf{x}, \mathbf{y})$ as a compatibility function that measures how well the output $\mathbf{y}$ matches the given input $\mathbf{x}$.

The specific form of Ψ depends on the nature of the problem and special cases will be discussed subsequently. Using natural language parsing as an illustrative example, f_w can be chosen to be isomorphic to a Probabilistic Context Free Grammar (PCFG) (see e.g. [14]). Each node in a parse tree $\mathbf{y}$ for a sentence $\mathbf{x}$ corresponds to grammar rule g_j, which in turn has a score w_j. All valid parse trees $\mathbf{y}$ (i.e. trees with a designated start symbol S as the root and the words in the sentence $\mathbf{x}$ as the leaves) for a sentence $\mathbf{x}$ are scored by the sum of the w_j of their nodes. This score can thus be written in the form of Eq. 1, where $\Psi(\mathbf{x}, \mathbf{y})$ denotes the histogram vector of how often each grammar rule g_j occurs in the tree $\mathbf{y}$. This is illustrated on the right-hand side of Figure 1. $h_{\mathbf{w}}(\mathbf{x})$ can be efficiently computed by finding the structure $\mathbf{y} \in \mathcal{Y}$ that maximizes $f_{\mathbf{w}}(\mathbf{x}, \mathbf{y})$ via the CKY algorithm (see e.g. [14]).

For training the weights $\mathbf{w}$ of the linear discriminant function, we generalize the standard SVM optimization problem as follows [1,10,20,21]. A similar formulation was independently proposed in [16].

OP 2 (Structural SVM (primal))

$$\min_{\mathbf{w},\xi} \; \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{n}\sum_{i=1}^{n} \xi_i$$
$$s.t. \; \forall \mathbf{y} \in \mathcal{Y} : \mathbf{w}^T(\Psi(\mathbf{x}_i, \mathbf{y}_i) - \Psi(\mathbf{x}_i, \mathbf{y})) \geq \Delta(\mathbf{y}_i, \mathbf{y}) - \xi_i$$

The objective is the conventional regularized risk used in SVMs. The constraints state that for each training example $(\mathbf{x}_i, \mathbf{y}_i)$ the score $\mathbf{w}^T\Psi(\mathbf{x}_i, \mathbf{y}_i)$ of the correct $\mathbf{y}_i$ must be greater than the score $\mathbf{w}^T\Psi(\mathbf{x}_i, \mathbf{y})$ of all incorrect $\mathbf{y}$ by a difference of $\Delta(\mathbf{y}_i, \mathbf{y})$. Δ is an application dependent loss function that measures how different the two structures $\mathbf{y}_i$ and $\mathbf{y}$ are. Intuitively, the larger the loss, the further should the score be away from that of the correct training label $\mathbf{y}_i$. ξ_i is a slack variable shared among constraints from the same example, since in general the problem is often not separable. Note that $\sum \xi_i$ is an upper bound on the training loss $\mathcal{R}_S^{\triangle}(h)$.

```
Input: $S = ((\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n))$, $C > 0$, $\epsilon > 0$.
$K = \emptyset$, $\mathbf{w} = 0$, $\xi = 0$
repeat
    – $K_{org} = K$
    – for $i$ from 1 to $n$
        • $\mathbf{y} = \mathrm{argmax}_{\mathbf{y} \in \mathcal{Y}} \left[\Delta(\mathbf{y}_i, \mathbf{y}) + \mathbf{w}^T \Psi(\mathbf{x}_i, \mathbf{y})\right]$ # find most violated constraint
        • if $\mathbf{w}^T(\Psi(\mathbf{x}_i, \mathbf{y}_i) - \Psi(\mathbf{x}_i, \mathbf{y})) < \Delta(\mathbf{y}_i, \mathbf{y}) - \xi_i - \epsilon$ # violated more than $\epsilon$?
            ∗ $K = K \cup \left\{\mathbf{w}^T(\Psi(\mathbf{x}_i, \mathbf{y}_i) - \Psi(\mathbf{x}_i, \mathbf{y})) \geq \Delta(\mathbf{y}_i, \mathbf{y}) - \xi_i - \epsilon\right\}$
            ∗ $(\mathbf{w}, \xi) = argmin_{\mathbf{w}, \xi} \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{n}\sum_{i=1}^{n} \xi_i$ subject to $K$.
until $(K = K_{org})$
Output: $\mathbf{w}$
```

Fig. 2. Cutting plane algorithm for training Structural SVMs

While the training problem is obviously still convex and quadratic, it typically has exponentially many constraints. For most choices of $\mathcal{Y}$ (e.g. sequences and trees), the cardinality of $\mathcal{Y}$ is exponential in the maximum size of $\mathbf{x}$ — and so is the number of constraints in OP2. This makes solving OP2 intractable using off-the-shelf techniques. However, it has been shown that the cutting plane algorithm in Figure 2 can be used to efficiently approximate the optimal solution of this type of optimization problem [21,12]. The algorithm starts with an empty set of constraints, adds the most violated constraint among the exponentially many during each iteration, and repeats until the desired precision $\epsilon > 0$ is reached. It can be proved that only a polynomial number of constraints will be added before convergence [21,12]. One crucial aspect of the algorithm, however, is the use of an oracle that can find the most violated constraint among the exponentially many possible constraints in polynomial time. That is, we need to compute

$$\underset{y \in \mathcal{Y}}{\mathrm{argmax}}[\Delta(\mathbf{y}_i, \mathbf{y}) + \mathbf{w}^T \Psi(\mathbf{x}_i, \mathbf{y})]. \tag{1}$$

For many $\mathcal{Y}$, feature mappings Ψ, and the loss functions Δ, this problem can be solved via dynamic programming. For trees, for example, the argmax in Eq. (1) can be computed using the CKY algorithm, if Ψ follows from a context-free grammar and Δ is any loss function that can be computed from the contingency table [21]. The running time of the overall learning algorithm is then polynomial in the number of training examples, the length of the sequences, and ϵ [21,12].

An alternative to the cutting plane algorithm is the algorithm proposed in [19]. It applies when the loss function Δ decomposes linearly and the argmax in Eq. (1) can be computed using a linear program that is guaranteed to have an integer solution.

4 Application Examples and Related Work

It has been shown for a range of application problems and structures $\mathcal{Y}$ that SVM training is feasible and benficial. The work in [20,21] shows how structural SVMs can be applied to natural language parsing, sequence alignment, taxonomic classification, and named-entity recognition. More work on highly expressive models for parsing is given in [17], and the use of structural SVMs for protein threading is described in [10,12]. An alternative approach to alignment is [18]. Work on sequence tagging for natural language problems and OCR is given in [16,1]. Image segmentation is addressed in [2]. While traditional generative training can and has been used for many structural prediction problems in the past, the studies mentioned above have repeatedly shown that discriminative training gives superior prediction performance.

Conditional Random Fields (CRFs) [13] are the most popular alternative discriminative training methods for structured prediction problems. Like large-margin approaches, they also have shown excellent performance on a variety of problems. Instead of optimizing a regularized empirical risk for a user-defined loss function like in the SVM approach, CRFs optimize a regularized conditional likelihood. While they can be applied to many of the problems mentioned above, there is little direct comparison between SVM and CRF training yet.

Other training approaches for structured models include the perceptron algorithm and reranking approaches [4,3,5]. The structural SVM approach extends these. A very different approach to structured prediction is proposed in [23], implementing the structured prediction as a multivariate regression problem after mapping the structures into Euclidian space.

5 Summary

This paper provides a short summary of methods for Support Vector Machine training with structured outputs. In particular, it shows how a cutting-plane method can be used to solve the training problem efficiently despite an exponential number of constraints. Pointers towards applications and further reading provide a starting point for further exploration of this area of research.

References

1. Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden Markov support vector machines. In *International Conference on Machine Learning (ICML)*, 2003.
2. D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, and G. Heitz. Discriminative learning of markov random fields for segmentation of 3d scan data. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.
3. M. Collins. Discriminative reranking for natural language parsing. In *International Conference on Machine Learning (ICML)*, 2000.
4. M. Collins. Discriminative training methods for Hidden Markov Models: Theory and experiments with perceptron algorithms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2002.
5. M. Collins and N. Duffy. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Conference of the Association for Computational Linguistics (ACL)*, 2002.

6. Corinna Cortes and Vladimir N. Vapnik. Support–vector networks. *Machine Learning Journal*, 20:273–297, 1995.
7. K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research (JMLR)*, 2:265–292, 2001.
8. R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis.* Cambridge University Press, 1998.
9. T. Finley and T. Joachims. Supervised clustering with support vector machines. In *International Conference on Machine Learning (ICML)*, 2005.
10. T. Joachims. Learning to align sequences: A maximum-margin approach. online manuscript, August 2003.
11. T. Joachims. A support vector method for multivariate performance measures. In *International Conference on Machine Learning (ICML)*, 2005.
12. T. Joachims, T. Galor, and R. Elber. Learning to align sequences: A maximum-margin approach. In B. Leimkuhler et al., editor, *New Algorithms for Macromolecular Simulation*, volume 49 of *LNCS*, pages 57–68. Springer, 2005.
13. J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, San Francisco, 2001. Morgan Kaufmann.
14. C. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing.* MIT Press, Cambridge, MA, 1999.
15. V. Ng and C. Cardie. Improving machine learning approaches to coreference resolution. In *Annual Meeting of the Assoc. for Comp. Linguistics (ACL)*, 2002.
16. B. Taskar, C. Guestrin, and D. Koller. Maximum-margin markov networks. In *Neural Information Processing Systems (NIPS)*, 2003.
17. B. Taskar, D. Klein, M. Collins, D. Koller, and C. Manning. Max-margin parsing. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2004.
18. B. Taskar, S. Lacoste-Julien, and D. Klein. A discriminative matching approach to word alignment. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2005.
19. Ben Taskar. *Learning Structured Prediction Models: A Large Margin Approach.* PhD thesis, Stanford University, 2004.
20. I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *International Conference on Machine Learning (ICML)*, 2004.
21. I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research (JMLR)*, 6:1453 – 1484, September 2005.
22. V. Vapnik. *Statistical Learning Theory.* Wiley, Chichester, GB, 1998.
23. J. Weston, O. Chapelle, A. Elisseeff, B. Schölkopf, and V. Vapnik. Kernel dependency estimation. In *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. MIT Press.