Advanced Structured Prediction

Editors:

Sebastian Nowozin Sebastian.Nowozin@microsoft.com
Microsoft Research
Cambridge, CB1 2FB, United Kingdom

Peter V. Gehler pgehler@tuebingen.mpg.de
Max Planck Insitute for Intelligent Systems
72076 Tübingen, Germany

Jeremy Jancsary jermyj@microsoft.com
Microsoft Research
Cambridge, CB1 2FB, United Kingdom

Christoph Lampert chl@ist.ac.at
IST Austria
A-3400 Klosterneuburg, Austria

This is a draft version of the author chapter.

The MIT Press
Cambridge, Massachusetts
London, England

1 Structured Learning from Cheap Data

Xinghua Lou xinghua.lou@gmail.com
Sloan-Kettering Institute
New York, USA

Marius Kloft kloft@cbio.mskcc.org
Sloan-Kettering Institute and Courant Institute
New York, USA

Gunnar Rätsch raetsch@cbio.mskcc.org
Sloan-Kettering Institute
New York, USA

Fred A. Hamprecht fred.hamprecht@iwr.uni-heidelberg.de
Ruprecht-Karls-Universität Heidelberg
Heidelberg, Germany

Supervised structured learning has made many important contributions to a large collection of real world problems. The underlying notion of learning a dependency function between complex, structured input and output has notably broadened the applicability of machine learning, particularly into sophisticated problems from computer vision, natural language processing, computational biology and speech recognition. However, one issue, particularly noticeable in practice, is the notoriously heavy demand for ground truth annotation. Though this is common for any supervised learning method, this aspect is particularly challenging for structured data. Firstly, each training sample can consist of a large number of random variables whose state has to be specified. Secondly, annotators also have to consider rules to yield valid output structure. Manually parsing large, complex structures is very expensive, which remains an obstacle for structured learning in practice.

This chapter addresses this issue by enabling structured learning with "cheap data" – data that requires less annotation than usual. We consider three distinct methods for this purpose. This first refers to structured learning from partial annotations, namely only a fraction of the complex structured output *requires annotation. Then, an active learning based approach is presented, which interactively guides users to annotate important parts of samples. The third alternative is an extension of transfer learning to structured data. This is particularly useful when one wishes to apply a trained model to a new dataset that is acquired in an alike experimental condition yet is distinct in some aspects. All of those methods are demonstrated on a challenging cell tracking problem and the results show a substantial reduction of annotation effort while maintaining the same quality of the trained model.*

1.1 Introduction

Structured output predictions can typically be represented in terms of a graph with both vertex and edge attributes. For instance, each vertex may be associated with a semantic category, or each edge may have a connection strength. Such graphs are typically the minimizer of an optimization problem that combines unary terms — such as the propensity of a specific vertex to belong to a certain category — with constraints, or with higher-order terms, that couple the predictions associated with each node or edge.

A relevant instance of structured output prediction is the object tracking problem. In a tracking-by-assignment approach, we have unary terms that seek to predict if two targets in subsequent time steps are in fact identical. If such predictions were made independently, the result may be paradoxical in that a single target at time t is simultaneously associated with two different predecessors at time $t-1$. Clearly, if the merging of targets can be ruled out in the application at hand, then only either one or the other association can be correct, but not both at the same time.

Such structured output prediction problems usually consist of energy terms whose parameters have to be estimated. This is made possible by supervised structured learning, which aims at *directly* optimizing the parameters such that the prediction model can reproduce the experts' annotation as accurately as possible. Structured learning has significantly broadened the applications of machine learning to many different fields (Figure 1.1). As all supervised training, a sufficient and representative training dataset is required, which, however, becomes a nontrivial issue for structured data. Firstly, unlike canonical classification or regression whose output is a single variable, structured model consists of many more variables *per sample* (for instance, a DNA sequence can be as long as millions, Figure 1.1B). Secondly, those variables are interdependent, subject to some rules that have to be accounted for when annotating the sample (for instance, context-free grammars in parsing, Figure 1.1A).

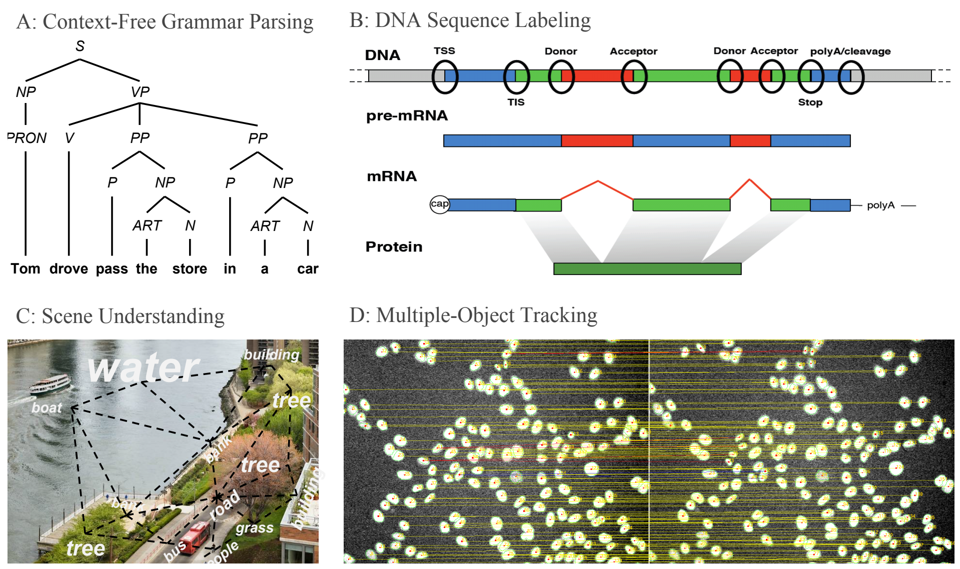

Figure 1.1: Typical structure prediction problems in natural language processing (A), computational biology (B) and computer vision (C and D).

This chapter is all about learning approaches that can make do with less human effort, which we refer to as *structured learning from cheap data*. We will use tracking-by-assignment as a running example in this chapter. Section 1.3 will show how it can be cast as a structured learning problem, opening the way to a principled parametrization of expressive models based on training data alone. However, in biological applications, we may easily observe thousands of targets in each frame of a video. A standard structured learning setup would consequently need training samples each of which has a very complex and large structure. Generating such expansive training data is tedious at best.

Section 1.4 shows how to *learn from partial annotations*. The un-annotated parts of the data are treated as latent variables that also need to be optimized over. This often leads to hard, non-convex optimization problems that can only be solved approximately by expectation-maximization type procedures. Computational efficiency is key in such iterative procedures, and we show how notable speed-ups can be achieved by the recycling of approximation bounds and an adjustment of convergence criteria over time. Given a good initialization (which is necessary given that these iterative schemes end up in a local optimum), such procedures reach competitive accuracy with only a fraction of full annotations. Theoretically, Section 1.4 also proves consistency of the loss functions used therein, and offers a probabilistic bound on the generalization error of structured learning from partial annotations.

Section 1.5 presents an attractive alternative based on *active learning*, where one iteratively identifies part of a training set that is deemed most informative. The rationale is that judiciously choosing the training examples to be labeled should afford steeper learning curves (accuracy as a function of training set size) than randomly selecting a subset for labeling.

Finally, in Section 1.6, we illustrate *structured transfer learning*. The idea here is to regularize the training procedure by coupling the learning of the parameters to a related but different learning problems for which abundance training data is already available. This technique is an embodiment of the notion “extra data for better regularization”.

The motivation, prior work and necessary notation will be introduced at the beginning of each of the three main sections. In Section 1.7 we conclude with a brief discussion of the presented and future work.

1.2 Short Abstract

Structured learning is a powerful paradigm. However, in its basic formulation, it requires fully annotated *and* accurate training data. Both requirements are often impractical, especially if training data needs to be generated by human experts. Several extensions of structured learning seek to relieve the annotators’ plight and make the learning more “convenient”. For the relevant problem of tracking an unknown number of divisible objects, we highlight in a tutorial manner (i) the formulation as a structured learning problem, (ii) structured learning from partial annotations, (iii) active structured learning and (iv) structured domain adaptation.

Keywords: cheap data, annotation cost, partial annotation, active learning, transfer learning, max-margin, bundle method

1.3 Running Example: Structured Learning for Cell Tracking

Before diving into the technical details, we first introduce a structured prediction model for cell tracking that we will use throughout this chapter.

1.3.1 Background

Unlike conventional computer vision problems such as surveillance analysis which contains a handful of (heterogeneous) objects, bio-image sequence normally contains hundreds and even thousands of homogeneous objects that are divisible according to some biological process (for instance, cell division). The combination of such vast amount and the very complex underlying temporal events raises a new challenge to the vision community. As discussed in Meijering et al. (2009), conventional tracking techniques are not applicable because of either limited expressive power (for instance, level-set) or low scalability (for instance, particle filter). Alternatively, *tracking-by-assignment* methods have shown promising performance in capturing complex mixture of events (Padfield et al., 2011) while also keep being scalable to even thousands of objects (Lou et al., 2011).

1.3.2 Generalized Pairwise Tracking Models

We assume a robust detection algorithm to detect objects (i.e., cells) but we accept errors such as over-segmentation and under-segmentation. We propose a *generalized pairwise tracking model* that encloses a mixture of events such as cell migration, cell division, as well as over-/under-segmentation, see Figure 1.2. Formally, given sets of detected objects $\{\boldsymbol{C},\, \boldsymbol{C}'\}$ from two subsequent frames, the model assumes a multitude of possible assignment hypotheses (e.g., events) and seeks a subset that is most compatible with the observations and with the parameter learned from the training data:

$$\underset{y\in\{0,1\}^{|y|}}{\operatorname{argmax}} \quad L(x,y;w) := \sum_{e\in\boldsymbol{E}} \sum_{c\in P(\boldsymbol{C})} \sum_{c'\in P(\boldsymbol{C}')} \langle f^e_{c,c'}, w^e\rangle y^e_{cc'} \tag{1.1}$$

$$\text{s.t.} \quad \forall c' \in P(\boldsymbol{C}'), \sum_{e\in\boldsymbol{E}} \sum_{c\in P(\boldsymbol{C})} y^e_{c,c'} = 1, \ \text{(consistency)} \tag{1.2}$$

$$\forall c \in P(\boldsymbol{C}), \sum_{e\in\boldsymbol{E}} \sum_{c'\in P(\boldsymbol{C}')} y^e_{c,c'} = 1. \ \text{(consistency)} \tag{1.3}$$

Here, $f^e_{c,c'}$ is a feature vector for the hypothetical event e between objects c and c', and, parametrized by w^e, $\langle f^e_{c,c'}, w^e\rangle$ is the linear scoring of this hypothesis, which is counted if $y^e_{cc'} = 1$ (i.e., selected). However, y is subject

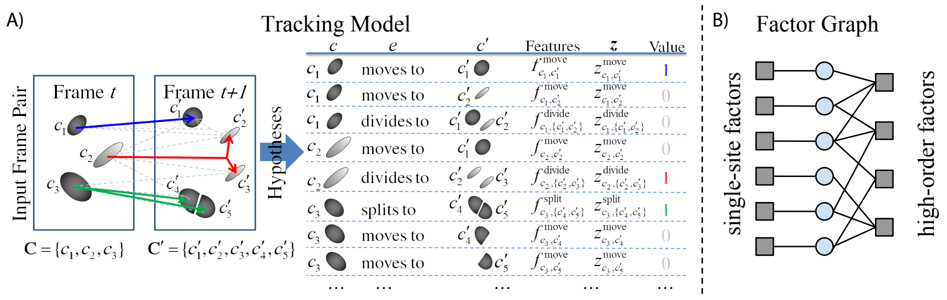

Figure 1.2: Panel A) Toy example: two sets of object candidates, and a small subset of the possible assignment hypotheses. One particular interpretation of the scene is indicated by solid arrows (left) or equivalently by a configuration of binary indicator variables y (rightmost column in table). Some rejected hypotheses are shown as light gray dash lines. Panel B) A factor graph representation of the proposed pairwise tracking model, which consists of unary potential as individual event scoring and high-order potential for guaranteeing consistency.

to consistency constraints: each candidate in the first frame must have a single fate, and each candidate from the second frame a unique past. That is, for hypotheses associated with the same candidate, only one of them can be accepted. To this end, as the corresponding factor graph representation (Kschischang et al., 2001) shows (Figure 1.2 B), this model consists of unary factors that represent the scoring of individual hypothetical events and high-order factors that couple those events and guarantee consistency.

Obviously, (1.1) is a linear model, as $L(x, y; w) := \langle w, \Phi(x, y)\rangle$. Here, w is the concatenation of event-specific parameters and $\Phi(x, y)$ is the concatenation of event-specific features summed up over all activated events, which is referred to as the *joint feature vector*.

For a given parameter w, we use integer linear programming (ILP) solvers to find the best assignments. Commercial solvers such as IBM's CPLEX or Gurobi's tools can scale up to thousands of hypotheses (Lou et al., 2011).

1.3.3 Max-Margin Formulation and Optimization

Given N training frame pairs $\boldsymbol{X} = \{x_n\}$ and their correct assignments $\boldsymbol{Y}^* = \{y_n^*\}$, $n = 1, \ldots, N$, we we attempt to find the decision boundary that maximizes the margin between the correct assignment y_n^* and the closest runner-up solution, i.e., the canonical max-margin learning paradigm (Taskar et al., 2003; Tsochantaridis et al., 2006):

$$\begin{aligned} \underset{w,\boldsymbol{\xi}\geq\mathbf{0}}{\operatorname{argmin}} \quad & \lambda\Omega(w) + \sum_{n=1}^{N} \xi_n \\ \text{s. t.} \quad & \forall n, \forall y \in Y_n, L(x_n, y_n^*; w) - L(x_n, y; w) \geq \Delta(y_n^*, y) - \xi_n, \end{aligned} \tag{1.4}$$

where Y_n is the output space and using $\Delta(y_n^*, \hat{y}_n)$ instead of a fixed margin is known as *margin rescaling* (Tsochantaridis et al., 2006).

Since (1.4) involves an exponential number of constraints, the learning problem cannot be represented explicitly, let alone be solved directly. We thus resort to the *bundle method* (Teo et al., 2010; Do and Artieres, 2012) which, in turn, is based on the *cutting plane* approach (see, for instance, Tsochantaridis et al. (2006); Joachims et al. (2009); Rätsch et al. (2002)). Briefly, bundle methods iteratively construct piece-wise linear bounds for the empirical loss (i.e., “cutting” planes) until the bounds are sufficiently tight. The procedure terminates when approximation gap ϵ, i.e., the difference between the true objective function and its linear bounds at current w, reaches a threshold (Teo et al., 2010).

1.3.4 Results: First Look

We compared the structured output learning algorithm above with L_1 and L_2 regularizer against several state-of-the-art cell tracking methods on the DCellIQ dataset provided by Li et al. (2010). The results can be found in Table 1.1. Our structured learning based method(s) outperforms all of the other methods with a clear margin. Compared to Li et al. (2010) who first studied this dataset, we obtained an improvement by more than one order of magnitude (0.30% *vs.* 6.18% loss), illustrating the power of structured output learning for this application.

1.3.5 Annotation Cost for Training Data Preparation

The encouraging performance boost by structured learning has a major requirement: a sufficiently large set of *representative training samples*. However, manually annotating hundreds of events per pair of frames is particu-

Table 1.1: Comparison of average loss using six approaches on the DCellIQ dataset. Compared to Li et al. (2010) who first studied this dataset, structured learning obtained an improvement by a factor of 20 (0.30% *vs.* 6.18% loss).

Method	**Description**	**Avg. Loss**
Li et al. (2010)	Graph matching, no learning	6.18%
Padfield et al. (2011)	ILP as max-flow, no learning	1.64%
Manual tweaking	Tweaking via visual inspection on results	1.12%
Random Forest	Learning local event classifiers	0.55%
Structured learning (L_1)	Eq. (1.4) with L_1 regularization	0.45%
Structured learning (L_2)	Eq. (1.4) with L_2 regularization	**0.30%**

larly labor-intensive and time-consuming. This severely limits the applicability of such advanced learning technique when to be deployed in real-world scenarios. For instance, annotating and validating a training dataset like the DCellIQ dataset takes **8 to 15 hours**.

1.4 Strategy I: Structured Learning from Partial Annotations

1.4.1 Motivation

Canonical structured learning always assume fully annotated data, i.e., specifying the state of each and every random variable in the structured output. This is particularly expensive for complex and/or large structured outputs. For example, in natural language processing manually constructing the entire parsing trees is labor-intensive. Also, in computational biology as our running example is, even a single sample (i.e., a pair of frames) contains hundreds of events. This motivates us to investigate the possibility of learning structured prediction models using only *partial* annotations, namely only a fraction of the complex structured output per training sample requires annotation. We consider this a viable approach because large, complex output structures are merely the compositional output of simple, local patterns. As per our examples above, the parsing tree is essentially constructed using rules from the context-free grammar, and, for cell tracking, the complete output assignment consists of local events such as cell migration and cell division.

We build on important previous work for multiclass classification with ambiguous labels. For example, Jin and Ghahramani (2002) proposed an EM-like algorithm that iteratively estimates the label distribution and classifies using this distribution as a prior. Recently, Cour et al. (2011) proposed convex loss for partial labels, which in turn resembles the one-versus-all loss (Zhang, 2004). We will extend this loss to structured data and discuss its properties. This work is also closely related to structured learning with latent variables (Yu and Joachims, 2009). The difference lies in the loss and the optimization strategy. Also, note that structured learning from partial annotations is different from semi-supervised or unsupervised structured learning (Zien et al., 2007). In those settings, training samples are either completely annotated or completely unannotated.

1.4.2 Formulation

Formally, we want to learn a structured prediction model from a *partially* annotated training set $\{(x_n, y_n^*) \in \mathcal{X}_n \times \mathcal{Y}_n\}_{n\in[N]}$. Here, x_n is a structured input from a space $\mathcal{X}_n$. (Note that the cardinality of the spaces $\mathcal{X}_n$, $\mathcal{Y}_n$ is typically different for each input n.) y^* is a *partially* annotated structured output which induces a partitioning of structured output space $\mathcal{Y}$ into two sets $\mathcal{Y}^* \cap \mathcal{Y}^\circ = \emptyset$, $\mathcal{Y}^* \cup \mathcal{Y}^\circ = \mathcal{Y}$. $\mathcal{Y}^*$ comprises all outputs that *are* compatible with a partial annotation y^*, while $\mathcal{Y}^\circ$ encompasses all those structured outputs that are *not* compatible with the partial annotation.

The Loss Structured learning needs to discriminate a correct structured output from an exponential number of wrong ones. We follow the max-margin argument (Tsochantaridis et al., 2006; Taskar et al., 2003) by constructing a loss function that penalizes small margins between the current prediction inferred from the partial annotation and the second best output, which, coupled with margin rescaling, leads to the following loss:

$$l^{\text{partial}}(x, y^*; \boldsymbol{w}) \quad = \quad \left| \max_{y \in \mathcal{Y}^{\text{P}}} [f(x, y; \boldsymbol{w}) + \Delta(y^*, y)] - \max_{y \in \mathcal{Y}^{\text{R}}} [f(x, y; \boldsymbol{w})] \right|_+$$

Here, $\mathcal{Y}^{\text{P}}$ is a "Penalty" space, since its members make a positive contribution to the loss. On the contrary, $\mathcal{Y}^{\text{R}}$ denotes a "Reward" space because it contains the correct configuration and brings a negative contribution. This loss resembles a generic structure for a number of other losses proposed in the literature (see a summary in Lou and Hamprecht (2012)). For example, if $\mathcal{Y}^{\text{P}} = \mathcal{Y}^\circ$ and $\mathcal{Y}^{\text{R}} = \mathcal{Y}^*$, $l^{\text{partial}}(x, y^*; \boldsymbol{w})$ becomes the **bridge loss** proposed in Lou and Hamprecht (2012), which is used in this chapter.

The Learning Objective Given the loss for partial annotations, we define the learning objective as

$$\min_{\boldsymbol{w}} \quad \lambda\Omega(\boldsymbol{w}) + \underbrace{\sum_n \max_{y \in \mathcal{Y}_n^{\text{P}}} [f(x_n, y_n; \boldsymbol{w}) + \Delta(y_n^*, y)]}_{P(\boldsymbol{w}),\ \text{convex}} - \underbrace{\sum_n \max_{y \in \mathcal{Y}_n^{\text{R}}} [f(x_n, y; \boldsymbol{w})]}_{R(\boldsymbol{w}),\ \text{convex}}$$
$$\text{s.t.} \qquad \text{each loss must be nonnegative.} \tag{1.5}$$

Equation 1.5 is a subtraction of two convex functions, namely $\lambda\Omega(\boldsymbol{w}) + P(\boldsymbol{w}) - R(\boldsymbol{w})$. Note that this formulation is equivalent to the canonical form with slack variables and (exponentially many) constraints. We keep this form to emphasize the structure of the objective which we will elaborate next. Note that, for max loss and bridge loss, the nonnegativity constraints can be achieved by ignoring the gradients of the samples that violate them during model update, as in usual SVM.

1.4.3 Optimization

In the sequel we will discuss the algorithmic aspects of solving (1.5).

The CCCP Algorithm The subtraction of two convex functions forms

a convex-concave optimization problem that can be solved by the CCCP procedure (Yuille and Rangarajan, 2003). Briefly, CCCP iterates between:
Step 1: At iteration t, estimate a linear upper bound on the concave function $-R(\boldsymbol{w})$ using its subgradient at $\boldsymbol{w}_t$, i.e., $\boldsymbol{v}_t = -\partial_{\boldsymbol{w}} R(\boldsymbol{w}_t)$.
Step 2: Update the model by $\operatorname{argmin}_{\boldsymbol{w}} \tilde{J}(\boldsymbol{w}) := \lambda\Omega(\boldsymbol{w}) + P(\boldsymbol{w}) + \langle \boldsymbol{v}_t, \boldsymbol{w} \rangle$.

Speeding Up CCCP with Bounds Recycling Structured learning is computationally expensive due to the repetitive maximization problems one has to solve at every iteration to compute the subgradients. This becomes even worse in the CCCP framework because a complete run of structured learning is performed largely from scratch per iteration. We now introduce a novel method for speeding up CCCP when structured learning is required.

The learning objective $\tilde{J}(\boldsymbol{w})$ in 1.5 has two important properties:
Complexity: $\tilde{J}(\boldsymbol{w})$ consists of three terms with different complexity: a regularizer $\lambda\Omega(\boldsymbol{w})$ (e.g., quadratic when using L2 regularization) and a linear term $\langle \boldsymbol{v}, \boldsymbol{w} \rangle$, both smooth and easy to solve, and a complicated, possibly non-smooth term $P(\boldsymbol{w})$.
Consistency: $\tilde{J}(\boldsymbol{w})$ changes at each CCCP iteration, due to the update of $\boldsymbol{v}$; however, the difficult function $P(\boldsymbol{w})$ remains the same.

These two observations lead to two ideas for speedup. Firstly, we construct a piece-wise linear lower bound on the difficult $P(\boldsymbol{w})$ only, rather than on the entire objective $\tilde{J}(\boldsymbol{w})$ as in Yu and Joachims (2009). Since the $P(\boldsymbol{w})$ part of $\tilde{J}(\boldsymbol{w})$ remains the same, we can reuse these bounds across multiple CCCP iterations and avoid recomputing them from scratch. When some "good" linear approximation for $P(\boldsymbol{w})$ is provided at each iteration, solving $\tilde{J}(\boldsymbol{w})$ is easy because the other two terms are simple. We name this technique *bounds recycling*, since the bounds will be reused to compute the approximation gap between the original objective and its linear approximation.

Secondly, CCCP iteratively matches points on the two convex functions (i.e., $\lambda\Omega(\boldsymbol{w})+P(\boldsymbol{w})$ and $R(\boldsymbol{w})$) which have the same subgradient, see Fig. 1.3 (left). Since we usually start with some $\boldsymbol{w}_0$ far from the optimum, it is not sensible to solve $\tilde{J}(\boldsymbol{w})$ to high precision at early iterations. Otherwise, many bounds need be computed to achieve this precision at some immature $\boldsymbol{w}$, which are mostly not reused at later iterations when precision really matters. Therefore, we propose to adaptively increase the precision of CCCP iteration until reaching the required precision. This procedure, named *adaptive precision*, is shown in Fig. 1.3 (right). Since the training precision is controlled by the approximate gap ϵ, this means we gradually decrease ϵ per CCCP iteration (see line 5, Algorithm 1.1).

Solving Model Update in the Dual To construct a lower bound approximation for $P(\boldsymbol{w})$, we follow the bundle minimization method from Teo et al.

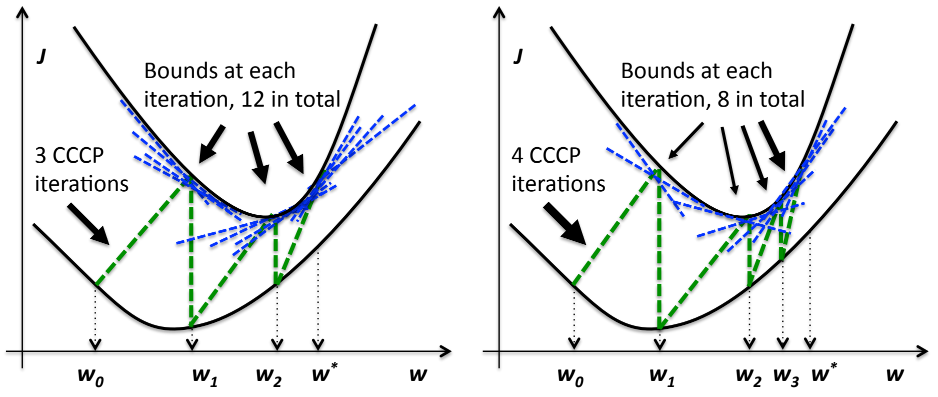
J
Bounds at each iteration, 12 in total
3 CCCP iterations
w0
w1
w2
w*
w
J
Bounds at each iteration, 8 in total
4 CCCP iterations
w0
w1
w2
w3
w*
w

Figure 1.3: CCCP procedure: starting from $\boldsymbol{w}_0$, iteratively match points in the two curves which have the same subgradient, until convergence to the optimal $\boldsymbol{w}^*$. CCCP with fixed precision (left) requires fewer iterations, but more bounds than CCCP with adaptive precision (right).

(2010). Briefly, at some $\boldsymbol{w}_t$, we compute the subgradient of $P(\boldsymbol{w})$ and the corresponding offset, denoted as $\boldsymbol{a}$ and b, respectively.

Now, this lower bound sitting at $\boldsymbol{w}_t$ can be expressed as $\langle \boldsymbol{a}, \boldsymbol{w}\rangle + b \leq P(\boldsymbol{w}), \forall \boldsymbol{w}$. We store all subgradients $\boldsymbol{a}$ as column vectors in $\boldsymbol{A} = [\boldsymbol{a}_0, \boldsymbol{a}_1, \ldots]$ and the offsets b in $\boldsymbol{b} = [b_0, b_1, \ldots]'$. Given $\boldsymbol{A}$ and $\boldsymbol{b}$, solving $\tilde{J}(\boldsymbol{w})$ in (1.5) becomes

$$\min_{\boldsymbol{w}} \quad \lambda\Omega(\boldsymbol{w}) + \underbrace{\max_{(\boldsymbol{a},b)\in(\boldsymbol{A},\boldsymbol{b})} (\langle \boldsymbol{a}, \boldsymbol{w}\rangle + b)}_{\text{Linearly lower bounded } P(\boldsymbol{w})} + \langle \boldsymbol{v}, \boldsymbol{w}\rangle \tag{1.6}$$

Given regularizer $\Omega(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|^2$, this can be solved in its dual by

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & -\frac{1}{2\lambda}\boldsymbol{\alpha}'\boldsymbol{A}'\boldsymbol{A}\boldsymbol{\alpha} + \left(\boldsymbol{b}' - \frac{1}{\lambda}\boldsymbol{v}'\boldsymbol{A}\right)\boldsymbol{\alpha} \\ \text{s.t.} \quad & \boldsymbol{\alpha}'\mathbf{1} = 1, \boldsymbol{\alpha} \geq \mathbf{0}. \end{aligned} \tag{1.7}$$

The primal variable $\boldsymbol{w}$ is connected to $\boldsymbol{\alpha}$ by $\boldsymbol{w} = -\frac{1}{\lambda}(\boldsymbol{v} + \boldsymbol{A}\boldsymbol{\alpha})$. It is possible collapse the previous lower bounds to a small number without loss of accuracy or convergence guarantees Do and Artieres (2012).

1.4.4 Theoretical Analysis

In this section, we show a generalization bound for the proposed method of structured learning with partially annotated outputs. This establishes the theoretical guarantee that the algorithm will not overfit, given sufficiently many training examples.

Theorem 1.1 (Generalization Bound for Structured Learning with Par-

Algorithm 1.1 Structured learning from partial annotations

```
1:  **Input:** $\{x_n, y_n^*\}$, $\boldsymbol{w}_0$, $\eta$, $\{\epsilon, \epsilon_{\min}, \rho\}$
2:  Initialize $t = 0, k = 0, \boldsymbol{A} = \emptyset, \boldsymbol{b} = \emptyset, \boldsymbol{w} = \boldsymbol{w}_0$
3:  **repeat**
4:      Compute $\boldsymbol{v}_t$ as the upper bound of the concave function
5:      Set adaptative precision $\epsilon = \max(\epsilon \times \rho, \epsilon_{\min})$
6:      **repeat**
7:          Compute gradient $\boldsymbol{a}_k$ and offset $b_k$
8:          Set $\boldsymbol{A} = \boldsymbol{A} \cup \boldsymbol{a}_k$ and $\boldsymbol{b} = \boldsymbol{b} \cup \boldsymbol{b}_k$
9:          Update $\boldsymbol{w}$ using Eq. 1.7 with $\boldsymbol{A}$, $\boldsymbol{b}$ and $\boldsymbol{v}_t$
10:         Compute approximation gap $\hat{\epsilon}$ (see Teo et al. (2010))
11:         Set $k = k + 1$
12:     **until** $\hat{\epsilon} \le \epsilon$
13:     Set $\boldsymbol{w}_{t+1} = \boldsymbol{w}$
14:     Set $t = t + 1$
15: **until** $\tilde{J}(\boldsymbol{w}_{t-1}) - \tilde{J}(\boldsymbol{w}_t) \le \eta$
16: **Output:** $\boldsymbol{w}$
```

tially Annotated Outputs). *Let $S = (x_n, y_n^*)_{1 \le n \le N}$ be an i.i.d. family of random variables with $y_n^* \in \mathcal{Y}^p \supset \mathcal{Y}$ such that there exist $B > 0$ such that $\mathbb{P}(\|\Phi(x, y)\|_{\le} B) = 1$. Let $\Delta^{\max} := \sup_{y,y'} \Delta(y, y')$. Put $l(x_n, y_n^*, \boldsymbol{w}) := \left|\max_{y \in \mathcal{Y}_n^\circ} (\langle \boldsymbol{w}, \Phi(x, y)\rangle + \Delta(y_n, y)) - \max_{y \in \mathcal{Y}_n^*} \langle \boldsymbol{w}, \Phi(x, y)\rangle\right|_+$. Denote $\boldsymbol{w}^* \in$* argmin$_{\boldsymbol{w}:\|\boldsymbol{w}\| \le \mu}$ $\mathbb{E}[l(x, y, \boldsymbol{w})]$ *and* $\widehat{\boldsymbol{w}}_N \in$ argmin$_{\boldsymbol{w}:\|\boldsymbol{w}\| \le \mu}$ $\widehat{\mathbb{E}}[l(x, y, \boldsymbol{w})]$. *Then, with probability at least $1 - \delta$, the generalization error of* structured prediction with partially annotated outputs *is bounded by:*

$$\mathbb{E}[l(x, y, \widehat{\boldsymbol{w}}_N)|S] - \mathbb{E}[l(x, y, \boldsymbol{w}^*)] \le \frac{(\mu B + \Delta^{\max})\left(8\,|\mathcal{Y}^p|\,|\mathcal{Y}| + \sqrt{2\log(2/\delta)}\right)}{\sqrt{N}}.$$

Due to the chapter space constraint, the detailed proof is available online at `http://raetschlab.org/suppl/mitbookstruct`. The proof follows similar ideas in multiclass classification (Koltchinskii and Panchenko, 2002). We observe that the bound depends quadratically on the size of the output space, which can be very large and render the value of the bound high. For specific structures such as hidden Markov models, it might be possible to obtain a tighter bound (cf. chapter 11 of Bakir et al. (2006)). The above bound establishes consistency in the sense that $\mathbb{E}[\widehat{\boldsymbol{w}}_N] - \mathbb{E}[\boldsymbol{w}^*] \to 0$, when $N \to \infty$. Another interesting question is whether the formulation fulfills consistency with respect to the discrete loss function Δ. Such an analysis was presented in McAllester and Keshet (2011), who showed asymptotic consistency of the update direction of a perception-like structured prediction algorithm. Whether such a result also holds for an analog perceptron-like algorithm using partially annotated labels is unknown at the present time.

1.4.5 Results

We use training data (DCellIQ from Li et al. (2010)) and test data (Mitocheck from `www.mitocheck.org`) from two different labs for a realistic demonstration.

Comparison to Structured Perceptron and Full Annotation On the running example, structured learning from partial annotations is compared against bundle method for risk minimization (Lou and Hamprecht, 2011) using full annotations, and structured perceptron with partial annotations (Fernandes and Brefeld, 2011). To make all experiments comparable, the same (training) precision, i.e. the approximation gap (Teo et al., 2010), was used for bundle method and the method proposed here. The structured perceptron with partial annotations was trained until the task loss, i.e. the true loss $\Delta(\cdot,\cdot)$, became zero, or stopped improving, using any early stopping.

Fig. 1.4 shows a comparison of the average test loss (specifically, the task loss $\Delta(\cdot,\cdot)$). Firstly, the tracking model trained using 25% partial annotation is comparable to a model training using fully annotated data. Secondly, the proposed method consistently outperforms the structured perceptron with partial annotation. We attribute this to the perceptron's lack of regularization, and resulting overfitting. Fig. 1.5 shows a comparison of training times. Once the proportion of partial annotation exceeds 20%, our method requires roughly twice as much time as the bundle method for risk minimization that is working on full annotations only. Training the structured perceptron appears to be more expensive.

Comparison of Optimization We compare our optimization strategy to the CCCP procedure from Yu and Joachims (2009) which does not use the

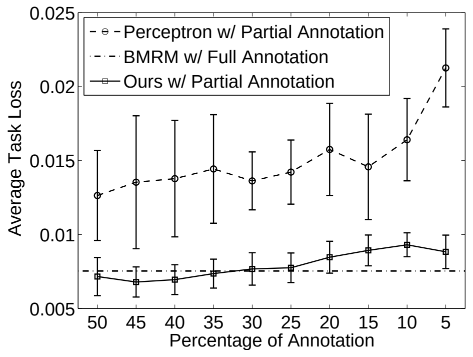

Figure 1.4: Comparison of average test loss. The model trained using 25% partial annotation is comparable to a model training using fully annotated data.

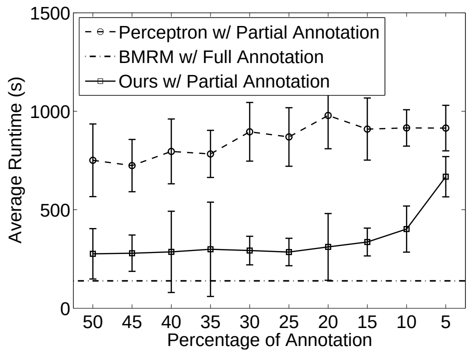

Figure 1.5: Comparison of training time. Training time is doubled compared to BMRM, yet this is much more affordable than the annotation cost.

bounds recycling and adaptive precision proposed here. We also study the effect of omitting either bounds recycling or/and adaptive precision.

Fig. 1.6 shows the convergence of the objective function. All optimization methods converge to the same objective value. Using both bounds recycling and adaptive precision, we achieve a speed-up of a factor of ≈ 5. Note that we implemented Yu and Joachims (2009)'s CCCP procedure using the BMRM method (Teo et al., 2010) whose complexity $\mathcal{O}(\frac{1}{\epsilon})$ is actually better than that of the proximal bundle method used in the original paper, $\mathcal{O}(\frac{1}{\epsilon^3})$. Fig. 1.7 shows the total number of bounds computed across the CCCP iterations. By using bounds recycling, our method only requires ca. 100 bounds until convergence, while Yu and Joachims (2009)'s approach computes almost 100 bounds at its first iteration.

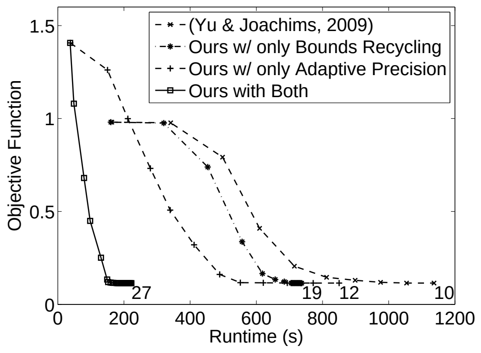

Figure 1.6: Decrease of the objective function. Using both bounds recycling and adaptive precision, we achieve a speed-up of a factor of ≈ 5 compared to Yu and Joachims (2009).

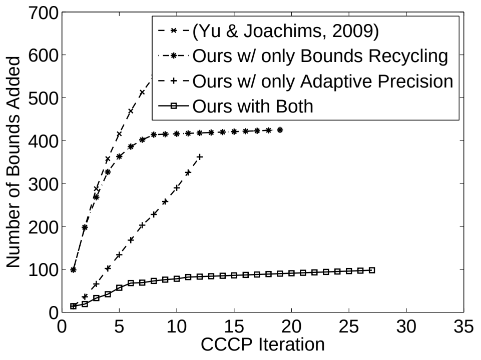

Figure 1.7: Total Number of bounds before convergence. We need ca. 100 bounds until convergence, while Yu and Joachims (2009) already computed almost 100 bounds at its first iteration.

1.5 Strategy II: Structured Data Retrieval via Active Learning

In the previous chapter we have described a strategy in which we can take advantage of partial annotation which is typically easier to obtain than a complete annotation. In many cases the annotation is produced in a manual effort. This section describes an alternative strategy based on *active learning* in which the annotator is guided through the dataset and asked to label only specific parts. This can lead to significant reductions of the labeling efforts.

1.5.1 Motivation

The concept of active learning is to guide users to annotate samples that are pivotal to improving the predictor and avoid wasting efforts on already well covered cases. One principled way is to estimate the uncertainty of each parameter in the model after structured learning, and then to identify that (part of a) training sample that will lead to the greatest reduction in uncertainty (Anderson and Moore, 2005). Unfortunately, such an endeavor is extremely costly (Krause and Guestrin, 2009) and not pursued here. Another good approach is to find that (part of a) training annotation whose inclusion in the training set minimizes the expected risk. Additional strategies of active learning for detecting rare positives were discussed in Warmuth et al. (2003). Even in unstructured prediction, such criteria are only tractable for specific classifiers such as Naive Bayes that allow efficient evaluation. In structured prediction, one possibility is to estimate the expected change of the predictions instead.

This chapter discusses a simple alternative, namely to break the large training instances into parts (a violation of their structure!) and to then identify those parts that look most informative, according to a variety of criteria. The parts selected by the algorithm can then be annotated by the human expert, added to the training set, etc. Our method consists of the following core components: uncertainty measure, model update and stopping criteria. In what follows, we elaborate on the details following the pseudocode shown in Algorithm 1.2.

Algorithm 1.2 Active structured learning with perceptron.

```
1:  Input: $\boldsymbol{D} = \{x_n\}_{n=1}^N, \boldsymbol{w}, \hat{\eta}, T$
2:  Initialize $\boldsymbol{D}_\mathrm{L} = \emptyset, \boldsymbol{D}_\mathrm{U} = \boldsymbol{D}, t = 1$
3:  repeat
4:      Find $\tilde{x} = \arg\max_{x \in \boldsymbol{D}_\mathrm{U}} q(x, \boldsymbol{w})$
5:      Annotate $\tilde{y}^*$
6:      Set $\boldsymbol{D}_\mathrm{U} = \boldsymbol{D}_\mathrm{U} \setminus \tilde{x}$
7:      Set $\boldsymbol{D}_\mathrm{L} = \boldsymbol{D}_\mathrm{L} \cup \{(\tilde{x}, \tilde{y})\}$
8:      for all $(x, y^*) \in \boldsymbol{D}_\mathrm{L}$ do
9:          Compute the best assignment $\hat{y}$
10:         Update $\boldsymbol{w} = \boldsymbol{w} + \Phi(x, y^*) - \Phi(x, \hat{y})$
11:     end for
12:     Compute average uncertainty $\bar{q}_t = \frac{1}{|\boldsymbol{D}_\mathrm{U}|} \sum_{x \in \boldsymbol{D}_\mathrm{U}} q(x, \boldsymbol{w})$
13:     Compute convergence measure $\eta(\bar{q}_{t-T:t})$ according to (1.8)
14:     Set $t = t + 1$
15: until $\eta(\bar{q}_{t-T:t}) \leq \hat{\eta}$ or $\boldsymbol{D}_\mathrm{U} \equiv \emptyset$
16: Output: $\boldsymbol{w}$
```

1.5.2 Algorithm

In this section we will discuss the algorithmic aspects of our approach.

Uncertainty Measure Proper means for measuring prediction uncertainty is vital to the uncertainty based active learning framework (Settles, 2012). We propose to use four different uncertainty measures described in Table 1.2. They are direct extensions of uncertainty measures for flat data (Tong and Koller, 2002; Schohn and Cohn, 2000) to structured data as in this paper. As line 4–6 of Algorithm 1.2 shows, at each iteration, we find the most uncertain sample (i.e., pair of patches) from all unlabeled samples $\boldsymbol{D}_{\mathrm{U}}$ and demand annotation from the annotator. We will compare the learning curves of those uncertainty measures in results.

Name	**Formulation and Description**
Random	$q(x, \boldsymbol{w}) \sim \mathrm{uniform}(0, 1)$
Scoring	$q(x, \boldsymbol{w}) = \exp\left(-\max_{y \in \mathcal{Y}} \boldsymbol{w}'\Phi(x, y)\right)$ Higher value of $\max_{y \in \mathcal{Y}} \boldsymbol{w}'\Phi(x, y)$ indicates higher confidence on the predicted tracking using existing parameter $\boldsymbol{w}$.
Best vs. Worst	$q(x, \boldsymbol{w}) = \exp\left(-\left(\max_{y \in \mathcal{Y}} \boldsymbol{w}'\Phi(x, y) - \min_{y \in \mathcal{Y}} \boldsymbol{w}'\Phi(x, y)\right)\right)$ Larger margin between those two terms indicates higher confidence towards the best predicted tracking w.r.t the worst one.
Best vs. 2nd	$q(x, \boldsymbol{w}) = \exp\left(-\left(\max_{y \in \mathcal{Y}} \boldsymbol{w}'\Phi(x, y) - \max_{y \in \mathcal{Y}^{\circ}} \boldsymbol{w}'\Phi(x, y)\right)\right)$ $\max_{y \in \mathcal{Y}^{\circ}} \boldsymbol{w}'\Phi(x, y)$ means computing the second best scoring and larger margin between those two terms indicates higher confidence towards the best predicted tracking w.r.t the second best one.

Table 1.2: List of uncertainty sampling strategy for comparison. *Random* assumes a uniform distribution of uncertainty on all samples. The rest are direct extensions of uncertain measures proposed in the literature on flat data (Tong and Koller, 2002; Schohn and Cohn, 2000).

Model Update At each iteration the model parameter $\boldsymbol{w}$ needs to be properly updated after receiving a newly annotated sample. Given a labeled training set $\boldsymbol{D}_{\mathrm{L}}$, a naïve way is to invoke max-margin structured learning from the previous Section 1.3.3. However, this turns out inefficient in practice: max-margin structured learning is known very expensive (Tsochantaridis et al., 2006), which means that the annotator has to wait a few minutes before proceeding to the next sample. Therefore, we resort to structured perceptron (Collins, 2002) for model update (line 8–11, Algorithm 1.2). Briefly, it makes a one-pass run through all labeled samples and updates the parameter by incrementally (and locally) adding the gradient, i.e., $\boldsymbol{w} = \boldsymbol{w} + \partial_{\boldsymbol{w}} \left(L(x, y^*; \boldsymbol{w}) - L(x, \hat{y}; \boldsymbol{w}) \right)$ (equivalent to line 10).

Stopping Criteria To decide when to terminate the entire active learning iteration, we chose a very popular measure proposed in Vlachos (2008) – the average uncertainty over all remaining unlabeled samples (cf. Table 1.2). This does not require any holdout validation dataset. At iteration t, given a sequence of computed average uncertainty $\bar{q}_{t-T:t}$ (incl. previous ones), we compute the convergence measure η from Laws and Schätze (2008) (line 12–13, Algorithm 1.2) using

$$\eta(\bar{q}_{1:T+1}) = |\widehat{\mathrm{mean}}(\bar{q}_{2:T+1}) - \widehat{\mathrm{mean}}(\bar{q}_{1:T})|, \tag{1.8}$$

where $\widehat{\mathrm{mean}}(\cdot)$ is the robust mean (i.e., mean of the elements within the 10% and 90% quantile). This convergence measure drops low when the improvement on average uncertainty remains minor for several iterations. We stop the active learning when the convergence measure is below a given threshold or all samples are labeled (line 15, Algorithm 1.2).

Combined Learning Strategy Though gaining speed, using structured perceptron for model update has two drawbacks: lack of regularization and local (thus noisy) gradient update (line 10, Algorithm 1.2). This makes the learned model prone to overfitting and also unstable in convergence. Therefore, in practice we use a combined approach: we use active structured perceptron *only* for training data retrieval and, after its convergence, we use max-margin structured learning to obtain a regularized and globally optimized model. It is also possible to call max-margin training multiple times during the active learning procedure, which is in spirit related to the hybrid training procedure in LASVM (Bordes et al., 2005).

Active Learning Complexity Analysis Assuming a pool of N unannotated samples, the complexity of the proposed Algorithm 1.2 is $\mathcal{O}(N^2)$. While the complexity of using dual max-margin structured learning for model update is at least $\mathcal{O}(N^3)$.

In Algorithm 1.2, at each iteration t $(1 \leq t \leq N)$ we need N predictions, among which $N-t$ predictions are for uncertainty estimation on unlabeled samples (line 4) and the rest t for model update using the newly labeled sample (line 8–11). This gives TN predictions after T iterations. Since T is a fraction of N, the overall complexity is $\mathcal{O}(N^2)$.

In the case of dual max-margin structured learning (cf. Section 1.3.3) for model update, Bottou (2007) shows that, in the dual SVM (and alike, e.g. max-margin structured learning), the number of support vectors scales at least linearly with the number of training samples. Thus, the complexity of max-margin structured learning using the dual is at least quadratic because we need compute the inner-product of each support vector and each sample. The complexity is at least $\sum_t \left[(N-t)+t^2\right]$, which amounts to $\mathcal{O}(N^3)$.

1.5.3 Results

We train on the DCellIQ dataset from Li et al. (2010) and test on the Mitocheck dataset. We first applied patchification on the training data (DCellIQ), namely a pair of full images is divided into pairs of local patches which are used for training. We consider patchification a necessary and viable pre-processing step for active learning. Otherwise, annotating a single sample with many patches is already too tedious and time-consuming, and part of the efforts is wasted on similar and repeated event patterns.

Uncertainty Measures and Stopping Criteria Using 660 patchified training samples from the DCellIQ dataset, in Fig. 1.8 we compare the learning curves (i.e., average uncertainty) of the four uncertainty measures up to 50% of the total training samples. *Best vs. Worst* is stably converging at the beginning but has a second wave of significant changes after 16% of total training samples. The same applies to *Scoring* but the changes of average uncertainty are more drastic. *Best vs. 2nd* appears to be the best performing one: it converges to a stable state after 17% of total training samples.

Regarding stopping criteria, *Random* is excluded because it is not suitable for the uncertainty convergence measure η according to (1.8). To compute η, we chose $T = 80$ and used 10^{-4} as the stopping threshold. As the embedded figure in Fig. 1.8 shows, they all stop at around 17% of training data.

To further understand the learning curve in a practical setting, we evaluated all intermediate $\boldsymbol{w}$ by the active learning on the test data, respectively for all uncertainty measures. The result in Fig. 1.9 further supports our choice of *Best 2nd* not only because of its superiority in stability but also for its lower test error.

Runtime In practice, using the structured perceptron for model update yields pleasant runtime. Across iterations it requires (stably) less than 9 seconds to perform model update and uncertainty computation. We consider this a tolerable delay for interactive labeling. Note that this runtime is dependent on the hardware specification of the computer because the underlying solver CPLEX can run the branch-and-bound ILP algorithm in parallel. We used a 2.40G Hz Intel Xeon machine with 12 cores.

Combined Learning We discussed the necessity of a follow-up max-margin training. Table 1.3 shows the result of this combined learning strategy (CL) using 17% (i.e., the stopping point by the convergence measure), 30% and 40% of training samples, compared against the active learning (AL) output.

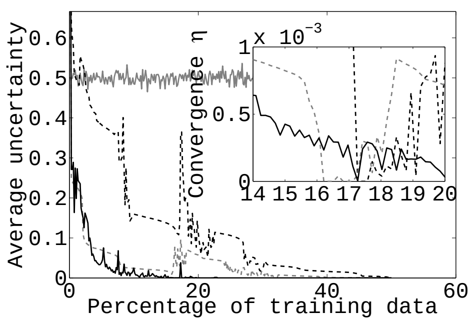

Figure 1.8: Comparison of uncertainty measures: average uncertainty *vs.* percentage of training data. The embedded figure shows the uncertainty convergence η *vs.* the percentage of training data.

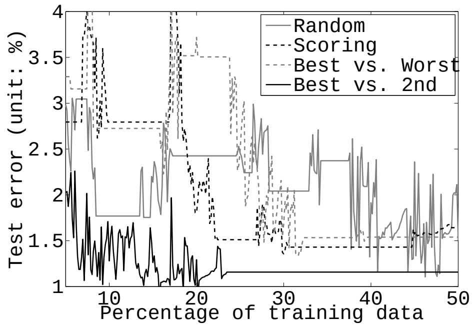

Figure 1.9: Comparison of uncertainty measure: test error *vs.* percentage of training data. *Best vs. 2nd* shows superior performance in terms of convergence speed and stability.

Table 1.3: Comparison of average test loss between active learning (AL) and combined learning (CL) on different percentages of training data. Among those uncertainty measures, *Best vs. 2nd* reaches a performance comparable to the baseline method (trained on all data) using 17% of the entire training data. The unit of the average loss is %.

Pct. of Training Data	17%		30%		40%	
Active or Combined Learning	AL	CL	AL	CL	AL	CL
Random	1.77	1.66	2.14	1.53	2.43	1.31
Scoring	3.72	1.78	2.79	1.73	1.80	1.11
Best vs. Worst	2.73	2.23	2.73	3.06	3.72	1.36
Best vs. 2nd	1.33	**1.08**	1.26	1.06	1.29	1.09
Baseline	1.07					

This affords the following observations. Firstly, using the same amount of training samples, regularized max-margin learning generally improves the output of active learning. Secondly, *Best vs. 2nd* performs better than the rest uncertainty measures. Finally (and most importantly), using *Best vs. 2nd* as uncertainty measure and using only 17% of the training samples, we can train a tracking model as competent as the baseline model learned from all samples (*baseline* in Table 1.3).

1.6 Strategy III: Structured Transfer Learning

1.6.1 Motivation

The previous strategies are designated for settings in which one has to construct training data completely from scratch. This section focuses on a different setting in which rich annotations are available for some datasets while we need to analyze another one that is different yet closely related. Typical examples include machine translation across similar languages and experimental data analysis with varying experimental conditions. Intuitively, we can reduce the extra effort on annotating the new dataset by exploiting its connection to those well annotated datasets. Such problems fall into the category of *transfer learning* which, in essence, is an embodiment of the notion "extra data for better regularization". This section presents an extension of transfer learning to structured data.

Transfer learning (Caruana, 1997; Evgeniou and Pontil, 2006) has been successfully applied to many real world problems such as sequence labeling in NLP (Pan and Yang, 2010) and mRNA splicing site recognition in computational biology (Schweikert et al., 2008) for a complete survey on this topic. For the particular case of structured data, Görnitz et al. (2011) considered transfer learning for hierarchical tasks for gene finding across species.

1.6.2 Formulation

Formally, we want to jointly learn from D datasets $\{\boldsymbol{D}^1, \ldots, \boldsymbol{D}^D\}$, where $\boldsymbol{D}^d = \{(x_n, y_n^{*d})\}_{n \in [N^d]}, d \in [D]$. A naïve approach is to to train on the union of all dataset, which is referred to as *Union*. Obviously, the *Union* formulation treat all datasets *identically* and the learning objective is to achieve a balanced performance over all datasets. This may help but is limited, particularly regarding the fact that datasets are not likely identical. To this end, we propose a second strategy that drops this condition. As Eq. (1.9) shows, we first regularize each dataset d using separate parameter vector $\boldsymbol{w}^d$ (left term). This avoids the "averaging" effect in *union*. Secondly, to still encode similarity between datasets, we add the middle regularization term that penalizes the difference between $\boldsymbol{w}^d$ and the shared component $\boldsymbol{w}$. The overall contribution of the middle term is controlled by ρ. After all, we introduced a higher degree of freedom to the model that allows to capture the similarity between datasets while respecting their distinction.

$$\min_{\substack{\boldsymbol{w},\boldsymbol{w}^1,\boldsymbol{w}^2,\dots\\ \boldsymbol{\xi}^1,\boldsymbol{\xi}^2,\dots}} \quad \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{\rho}{2}\sum_{d=1}^{D}\|\boldsymbol{w}^d - \boldsymbol{w}\|^2 + \sum_{d=1}^{D}\sum_{n=1}^{N^d}\xi_n^d \tag{1.9}$$

s.t.

$$\forall n \in [N^1], \forall y \in \mathcal{Y}_n^1, \langle \Psi(x_n^1, y_n^{*1}, y), \boldsymbol{w}^1\rangle \geq \Delta(y_n^{*1}, y) - \xi_n^1$$
$$\vdots$$
$$\forall n \in [N^D], \forall y \in \mathcal{Y}_n^D, \langle \Psi(x_n^D, y_n^{*D}, y), \boldsymbol{w}^D\rangle \geq \Delta(y_n^{*D}, y) - \xi_n^D$$

Eq. (1.9) is a quite generic and resembles several different strategies when parametrized accordingly. When $\rho = 0$, each $\boldsymbol{w}^d$ is learned for dataset d independently, which means no learning across datasets. When $\rho = \infty$, all $\boldsymbol{w}^d$ are forced to be identical to each other, which exactly leads to *Union*.

1.6.3 Optimization

For the *union*, the learning objective can be solved using bundle method (cf. Section 1.3.3). For the more complicated *transfer learning*, we provide an extension of the bundle method (Teo et al., 2010). Briefly, like in max-margin structured learning, we iteratively construct piece-wise linear lower bounds for the empirical loss, respectively for each dataset (i.e., domain). This leads to the following update rule for $\boldsymbol{w}$:

$$\min_{\boldsymbol{w},\boldsymbol{w}^1,\boldsymbol{w}^2,\dots} \quad \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{\rho}{2}\sum_{d=1}^{D}\|\boldsymbol{w}^d - \boldsymbol{w}\|^2 + \sum_{d=1}^{D} \max_{(\boldsymbol{a},b)\in(\boldsymbol{A}^d,\boldsymbol{B}^d)}\{\langle \boldsymbol{a}, \boldsymbol{w}^d\rangle + b\}$$

where $(\boldsymbol{A}^d, \boldsymbol{B}^d)$ denote the set of gradients and offsets which form the linear lower bounds for the empirical loss of domain d.

Using Lagrange multipliers, we can eventually obtain the dual form for the above formulation:

$$\max_{\boldsymbol{\alpha}^1,\boldsymbol{\alpha}^2,\dots} \quad -\frac{1}{2}\begin{bmatrix}\boldsymbol{\alpha}^1\\ \vdots \\ \boldsymbol{\alpha}^D\end{bmatrix}' \begin{bmatrix}\frac{1}{\tau}\boldsymbol{Q}^{11} & \dots & \frac{1}{\lambda}\boldsymbol{Q}^{1D}\\ \vdots & \vdots & \vdots\\ \frac{1}{\lambda}\boldsymbol{Q}^{D1} & \dots & \frac{1}{\tau}\boldsymbol{Q}^{DD}\end{bmatrix} \begin{bmatrix}\boldsymbol{\alpha}^1\\ \vdots \\ \boldsymbol{\alpha}^D\end{bmatrix} + \begin{bmatrix}B^1\\ \vdots \\ B^2\end{bmatrix}' \begin{bmatrix}\boldsymbol{\alpha}^1\\ \vdots \\ \boldsymbol{\alpha}^D\end{bmatrix}$$

$$\text{s.t.} \quad \forall d \in 1,\dots,D, \|\boldsymbol{\alpha}^d\|_1 \leq 1 \text{ and } \boldsymbol{\alpha}^d \geq 0.$$

Here, $\boldsymbol{Q}^{ij} = (A^i)'A^j$ and $\tau = \frac{\rho\lambda}{\rho+\lambda}$. Furthermore, the primal variables and the dual variables are connected by

$$\boldsymbol{w} = -\frac{1}{\lambda}\sum_{d=1}^{D} \boldsymbol{A}^d \boldsymbol{\alpha}^d \text{ and } \boldsymbol{w}^d = \boldsymbol{w} - \frac{1}{\rho}\boldsymbol{A}^d \boldsymbol{\alpha}^d, \forall d \in 1, \ldots, D. \tag{1.10}$$

1.6.4 Results

We experimented on the same DCellIQ and Mitocheck dataset. Our objective was to leverage the fully annotated DCellIQ data (e.g., *source*, 1188 samples) to ease the training of a model for the Mitocheck data (e.g., *target*, assumed newly acquired and lacking annotation, 2166 samples for training). The test data is a hold-out dataset sampled from Mitocheck (2165 samples). We compared five different learning strategies, as given in Table 1.4. As Figure 1.10 shows, when annotation in the target increases, *Transfer* converges to the baseline strategy (*All Target*) faster than the rest methods, and achieved a comparable performance at 20% target annotation. Afterwards, *Transfer* show very similar performance to *Union*, and they both outperform *All Target* after adding 30% target annotation which is an indication of the advantage of leveraging extra data for better regularization.

Table 1.4: Comparison of learning strategies – Unit %. The approximate gap parameter ϵ (see Teo et al. (2010)) is set to 10^{-2} throughout all strategies. The other parameters are selected using cross validation.

Strategy	**Formulation**	**Parameters**	**Trained on**
All Source	BMRM	$\lambda = 1$	All DCellIQ
All Target	BMRM	$\lambda = 1$	All Mitocheck
Partial Target	BMRM	$\lambda = 2.5$	Partial Mitocheck
Union	BMRM	$\lambda = 1$	All DCellIQ & partial Mitocheck
Transfer	Eq. (1.9)	$\lambda = 0.5$, $\rho = 2.5$	All DCellIQ & partial Mitocheck

1.7 Discussion and Conclusions

We have explored three different approaches for training structured prediction models with significant less annotations while maintaining a similar generalization performance.

We have proposed and theoretically as well as experimentally analyzed a method for structured output learning based on partial annotations. In many

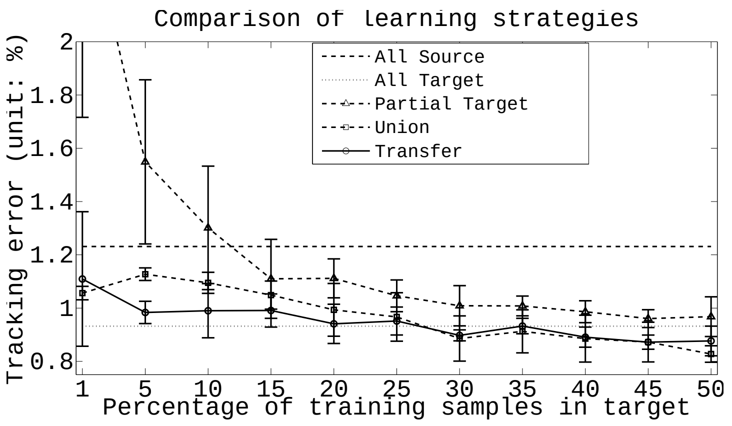

Figure 1.10: Comparison of learning strategies listed in Table 1.4. *Transfer* converges to the baseline strategy (*All Target*) faster than the rest methods, and achieved a comparable performance at 20% target annotation.

cases it is much easier to annotate only a part of an image or a sequence or to provide incomplete information about the structure. We proposed a novel algorithm based on bundle methods for solving a CCCP problem. Theoretically, we have shown that the proposed algorithm is consistent, and provided its generalization bound. Our experimental results show that we only need a tiny fraction ($\approx 5\%$) of the complete label information in order to achieve almost the same generalization performance as with full labels.

We have described and proposed two additional strategies for the same purpose. First, we considered a hybrid active learning strategy in which the algorithm quickly performs prediction and estimates its prediction uncertainty of many yet unlabeled patches. The annotator then iteratively labels the most uncertain patches. We have analyzed a few estimators for uncertainty and have shown that the best *vs.* 2nd best predictor performs best in our experiments. With less then 10% of the labeled training data the active learning algorithm predicts almost as well as with the full training data. We have also shown work on using transfer learning to reuse model information from prior experiments to train more accurate models with limited information in a new setting. Again with only $\approx 5\%$ of the data in the target domain, the accuracy is almost as good as with all data.

Depending on the prediction problem at hand and the specific difficulties of obtaining annotation data, different combinations of the presented and above-mentioned methods will lead to the best results. What we have described is a set of essentially orthogonal strategies of how to deal with costly annotations in practice.

1.8 References

B. Anderson and A. Moore. Active learning for hidden markov models: Objective functions and algorithms. In *International Conference on Machine Learning (ICML)*, 2005.

G. Bakir, T. Hofmann, B. Schoelkopf, A. J. Smola, B. Taskar, and S. Vishwanathan. *Predicting Structured Data*. MIT Press, Cambridge, MA, 2006.

A. Bordes, S. Ertekin, J. Weston, and L. Bottou. Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research*, 6:1579–1619, 2005.

L. Bottou. *Large-Scale Kernel Machines*. The MIT Press, 2007.

R. Caruana. Multitask learning. *Machine Learning*, 28:41–75, 1997.

M. Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Meeting of the Association for Computational Linguistics (ACL)*, 2002.

T. Cour, B. Sapp, and B. Taskar. Learning from Partial Labels. *Journal of Machine Learning Research*, 12:1225–1261, 2011.

T.-M.-T. Do and T. Artieres. Regularized bundle methods for convex and non-convex risks. *Journal of Machine Learning Research*, 2012.

A. A. T. Evgeniou and M. Pontil. Multi-task feature learning. In *Neural Information Processing Systems (NIPS)*, 2006.

E. Fernandes and U. Brefeld. Learning from partially annotated sequences. In *European Conference on Machine Learning (ECML)*, 2011.

N. Görnitz, C. Widmer, G. Zeller, A. Kahles, S. Sonnenburg, and G. Rätsch. Hierarchical multitask structured output learning for large-scale sequence segmentation. In *Advances in Neural Information Processing Systems*, 2011.

R. Jin and Z. Ghahramani. Learning with Multiple Labels. In *Neural Information Processing Systems (NIPS)*, 2002.

T. Joachims, T. Finley, and C.-N. Yu. Cutting-Plane Training of Structural SVMs. *Machine Learning*, 77(1):27–59, 2009.

V. Koltchinskii and D. Panchenko. Empirical margin distributions and bounding the generalization error of combined classifiers. *Annals of Statistics*, 30(1):1–50, 2002.

A. Krause and C. Guestrin. Optimal Value of Information in Graphical Models. *Journal of Artificial Intelligence Research*, 35:557–591, 2009.

F. R. Kschischang, B. J. Frey, and H. A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.

F. Laws and H. Schätze. Stopping criteria for active learning of named entity recognition. In *International Conference on Computational Linguistics (COLING)*, 2008.

F. Li, X. Zhou, J. Ma, and S. Wong. Multiple Nuclei Tracking Using Integer Programming for Quantitative Cancer Cell Cycle Analysis. *IEEE Transactions on Medical Imaging*, 29(1):96, 2010.

X. Lou and F. A. Hamprecht. Structured Learning for Cell Tracking. In *Neural Information Processing Systems (NIPS)*, 2011.

X. Lou and F. A. Hamprecht. Structured Learning from Partial Annotations. In

International Conference on Machine Learning (ICML), 2012.

X. Lou, F. O. Kaster, et al. DELTR: Digital Embryo Lineage Tree Reconstructor. In *IEEE International Symposium on Biomedical Imaging: From Nano to Macro (ISBI)*, 2011.

D. McAllester and J. Keshet. Generalization Bounds and Consistency for Latent Structural Probit and Ramp Loss. In *Neural Information Processing Systems (NIPS)*, 2011.

E. Meijering, O. Dzyubachyk, I. Smal, and W. A. van Cappellen. Tracking in cell and developmental biology. *Seminars in Cell & Developmental Biology*, 20(8): 894 – 902, 2009. ISSN 1084-9521.

D. Padfield, J. Rittscher, and B. Roysam. Coupled minimum-cost flow cell tracking for high-throughput quantitative analysis. *Medical Image Analysis*, 15(4):650–668, 2011.

S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

G. Rätsch, A. Demiriz, and K. Bennett. Sparse regression ensembles in infinite and finite hypothesis spaces. *Machine Learning*, 48(1-3):189–218, 2002.

G. Schohn and D. Cohn. Less is more: Active learning with support vector machines. In *International Conference on Machine Learning (ICML)*, 2000.

G. Schweikert, C. Widmer, B. Schölkopf, and G. Rätsch. An empirical analysis of domain adaptation algorithms for genomic sequence analysis. In *Advances in Neural Information Processing Systems (NIPS)*, 2008.

B. Settles. *Active Learning*. Morgan & Claypool, 2012.

B. Taskar, C. Guestrin, and D. Koller. Max-Margin Markov Networks. In *Neural Information Processing Systems (NIPS)*, 2003.

C. H. Teo, S. V. N. Vishwanthan, A. J. Smola, and Q. V. Le. Bundle methods for regularized risk minimization. *Journal of Machine Learning Research*, 11: 311–365, 2010.

S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *The Journal of Machine Learning Research*, 2:45–66, 2002.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large Margin Methods for Structured and Interdependent Output Variables. *Journal of Machine Learning Research*, 6(2):1453, 2006.

A. Vlachos. A stopping criterion for active learning. *Computer Speech & Language*, 22(3):295–312, 2008.

M. Warmuth, G. Rätsch, M. Mathieson, J. Liao, and C. Lemmen. Active learning in the drug discovery process. In *Advances in Neural Information Processing Systes (NIPS)*, 2003.

C. N. J. Yu and T. Joachims. Learning Structural SVMs with Latent Variables. In *International Conference on Machine Learning (ICML)*, 2009.

A. L. Yuille and A. Rangarajan. The Concave-Convex Procedure. *Neural Computation*, 15(4):915–936, 2003.

T. Zhang. Statistical Analysis of Some Multi-category Large Margin Classification Methods. *Journal of Machine Learning Research*, 5:1225–1251, 2004.

A. Zien, U. Brefeld, and T. Scheffer. Transductive support vector machines for structured variables. In *International Conference on Machine Learning (ICML)*, 2007.