# 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 H. Lampert** chl@ist.ac.at
*IST Austria*
*A-3400 Klosterneuburg, Austria*

The MIT Press
Cambridge, Massachusetts
London, England

# 1 Optimizing the Measure of Performance

**Joseph Keshet** joseph.keshet@biu.ac.il
*Bar-Ilan University*
*Ramat-Gan, Israel*

*The ultimate objective of discriminative learning is to train a system to optimize a desired measure of performance. In binary classification we are interested in finding a function that assigns a binary label to a single object, and minimizes the error rate (correct or incorrect) on unseen data. In structured prediction, we are interested in the prediction of a structured label, where the input is a complex object. Typically, each structured prediction task has its own measure of performance or evaluation metric, such as word error rate in speech recognition, the BLEU score in machine translation or the intersection-over-union score in object segmentation. In the chapter we review different objective functions for structured prediction and analyze them in the light of how they optimize to the desired measure of performance.*

Keywords: structured prediction; evaluation metric; measure of performance; support vector machines; conditional random fields; direct loss minimization; Perceptron

## 1.1 Introduction

We begin by posing the structured learning setting. We consider a supervised learning setting with input objects $\boldsymbol{x} \in \mathcal{X}$ and target labels $\boldsymbol{y} \in \mathcal{Y}$. The labels may be sequences, trees, grids, or other high-dimensional objects with internal structure. We assumed a fixed mapping $\boldsymbol{\phi} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$ from the set of input objects and target labels to a real vector of length $d$. We call the elements of this mapping *feature maps* or *feature functions*.

Here we consider a linear prediction rule with parameters $\boldsymbol{w} \in \mathbb{R}^d$, such that $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$ is a good approximation to the true label of $\boldsymbol{x}$, as follows:

$$\hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x}) = \underset{\boldsymbol{y}\in\mathcal{Y}}{\operatorname{argmax}}\ \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x},\boldsymbol{y}) \tag{1.1}$$

Ideally, we would like to find $\boldsymbol{w}$ such that the prediction rule optimizes the expected desired *measure of preference* or *evaluation metric* on unseen data. For example, phone error rate or word error rate in automatic speech recognition, intersection-over-union (PASCAL VOC) in object segmentation, NDCG in search engines and BLEU score in machine translation. We define a *cost* function, $\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}})$, to be a non-negative measure of error when predicting $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$ instead of $\boldsymbol{y}$ as the label of $\boldsymbol{x}$. Our goal is to minimize this function. Often the desired evaluation metric is a utility function that needs to be maximized (like BLEU, NDCG) and then we define the cost to be 1 minus the evaluation metric.

We assume that there exists some unknown probability distribution $\rho$ over pairs $(\boldsymbol{x},\boldsymbol{y})$ where $\boldsymbol{y}$ is the desired output (or reference output) for input $\boldsymbol{x}$. We then want to set $\boldsymbol{w}$ so as to minimize the expected cost, or the *risk*, for predicting $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$,

$$\boldsymbol{w}^* = \underset{\boldsymbol{w}}{\operatorname{argmin}}\ \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\rho}[\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x}))]. \tag{1.2}$$

This objective function is hard to minimize directly since the distribution $\rho$ is unknown. We use a training set $\mathcal{S}$ of $m$ examples that are drawn i.i.d. from $\rho$, $\mathcal{S} = \{(\boldsymbol{x}_1,\boldsymbol{y}_1),\ldots,(\boldsymbol{x}_m,\boldsymbol{y}_m)\}$. We replace the expectation in (1.2) with a mean over the training set. To avoid overfitting of the parameters $\boldsymbol{w}$ to the training set and to generalize over unseen test data, we add a normalization factor $\|\boldsymbol{w}\|_2^2$ that should reduce the capacity, or the VC dimension, of the parameters $\boldsymbol{w}$ (Vapnik, 2000).

The cost is often a combinatorial non-convex quantity, which is hard to minimize. Hence, instead of minimizing the cost directly, we minimize a lightly different function called a *surrogate loss*, denoted $\bar{\ell}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{y})$, and closely related to the cost. Overall the objective function of (1.2) transforms into the following objective function:

$$\boldsymbol{w}^* = \underset{\boldsymbol{w}}{\operatorname{argmin}}\ \frac{1}{m}\sum_{i=1}^{m} \bar{\ell}(\boldsymbol{w},\boldsymbol{x}_i,\boldsymbol{y}_i) + \frac{\lambda}{2}\|\boldsymbol{w}\|^2, \tag{1.3}$$

where $\lambda$ is a trade-off parameter between the loss term and the regularization.

One way to formalize what is meant by saying that a learning algorithm is able to learn and optimize a cost function is the notion of *strong consistency.* This notion requires that for all distributions $\rho$ on $(\boldsymbol{x},\boldsymbol{y})$, and for any feature map (finite or infinite dimensional), the weight vectors $\boldsymbol{w}^*$ produced by (1.3) satisfies

$$\lim_{m\to\infty} \frac{1}{m} \sum_{i=1}^{m} \bar{\ell}(\boldsymbol{w}^*, \boldsymbol{x}_i, \boldsymbol{y}_i) = \inf_{\boldsymbol{w}} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\rho}[\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x}))], \tag{1.4}$$

with probability one over the draw of the infinite sample.

We now review the different approaches to estimating the parameters $\boldsymbol{w}$, each with a different surrogate loss function.

## 1.2 Structured Perceptron

The seminal paper of Collins (2002) set the framework of structured prediction. It proposed an extension to the binary Perceptron algorithm to handle part-of-speech tagging. Perceptron is an online algorithm that works in rounds. At each round the algorithm receives an instance $\boldsymbol{x}$ and predicts a label $\hat{\boldsymbol{y}}$. Then the target label is given, $\boldsymbol{y}$, and the algorithm updates the weight vector $\boldsymbol{w}$. Starting with $\boldsymbol{w}_0 = \boldsymbol{0}$, after each round the algorithm outputs a weight vector $\boldsymbol{w}_t$. The weight vector after round $t$, $\boldsymbol{w}_{t+1}$, is defined as follows where $(\boldsymbol{x}_t, \boldsymbol{y}_t)$ is drawn from the distribution $\rho$

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \boldsymbol{\phi}(\boldsymbol{x}_t, \boldsymbol{y}_t) - \boldsymbol{\phi}(\boldsymbol{x}_t, \hat{\boldsymbol{y}}_{\boldsymbol{w}_t}) \tag{1.5}$$

Note that whenever $\hat{\boldsymbol{y}}_{\boldsymbol{w}_t} = \boldsymbol{y}_t$ then no update is made and we have $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t$. If $\hat{\boldsymbol{y}}_{\boldsymbol{w}_t} \neq \boldsymbol{y}_t$ then the update changes the parameter vector $\boldsymbol{w}_{t+1}$ in a way that favors $\boldsymbol{y}_t$ over $\hat{\boldsymbol{y}}_{\boldsymbol{w}_t}$.

In the *online* setting only the weight vector of each round $\boldsymbol{w}_t$ and its performance on the next (unseen) object $\boldsymbol{x}_{t+1}$ are of interest (there is no "test set"). In this chapter, however, we are interested in the *batch* setting, that is, in a single weight vector $\boldsymbol{w}^*$ that perform well on unseen data sampled from $\rho$.

The Perceptron algorithm is often described as a technique to solve a linear feasibility problem, that is aimed at satisfying a set of linear constrains (with no objective). Each constraint corresponds to a training example and asserts its correct prediction. If there is a weight vector $\boldsymbol{w}^*$ that satisfies all constraints, it can be shown that running iteratively over the data and update the weight vector $\boldsymbol{w}_{t+1}$ using (1.5) converges, and the number of updates is finite (Collins, 2002). In that case we can use the weight vector of the last round $\boldsymbol{w}_T$ to serve as $\boldsymbol{w}^*$, and we say that the data is linearly *separable*. When there no such $\boldsymbol{w}^*$ exists, we can convert the online Perceptron to a batch algorithm by setting $\boldsymbol{w}^*$ to the average of the weight vectors over all iterations (Dekel et al., 2004; Cesa-Bianchi et al., 2004).

We can think of the Perceptron algorithm as a stochastic sub-gradient descent (SSGD) solution of the following optimization problem[1]

$$\boldsymbol{w}^* = \text{argmin } \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\rho}\left[\boldsymbol{w}^\top\left(\boldsymbol{\phi}(\boldsymbol{x}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x})) - \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y})\right)\right]. \tag{1.6}$$

Using SSGD with a constant learning rate brings us to the update rule of (1.5). This optimization problem (1.6) is non-convex. From our perspective, however, a bigger problem is that the optimization problem defined by (1.6) is quite different from (1.2): the optimization problem (1.6) is defined independently of the cost function and therefore cannot be expected to yield good solutions to (1.2) when the data is not linearly separable.

## 1.3 Large Margin Structured Predictors

The idea is to generalize the hinge loss function used in binary support vector machines (SVMs) to the structured case. The first formulation was introduced by Taskar et al. (2003) and it is called *max-margin Markov networks (M$^3$)*, where the generalized surrogate loss function is expressed in terms of the Hamming distance between the target label $\boldsymbol{y}$ and the predicted label $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$. Denote by $H(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}})$ the Hamming distance between $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$. The M$^3$ can be formulated by using the following *Hamming hinge loss* as a surrogate loss in the optimization function (1.3):

$$\bar{\ell}_{hamming}(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y}) = \max_{\hat{\boldsymbol{y}}\in\mathcal{Y}}\left[H(\boldsymbol{y}, \hat{\boldsymbol{y}}) - \boldsymbol{w}^\top\boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}) + \boldsymbol{w}^\top\boldsymbol{\phi}(\boldsymbol{x}, \hat{\boldsymbol{y}})\right] \tag{1.7}$$

The use of Hamming hinge loss function can be motivated as being a direct extension of the binary SVM. Consider the case where $\mathcal{Y} = \{-1, +1\}$ so $y \in \mathcal{Y}$ is a scalar, and $y = -\hat{y}_{\boldsymbol{w}}$. Set $\phi(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{2}y\boldsymbol{\psi}(\boldsymbol{x})$, where $\boldsymbol{\psi} : \mathcal{X} \to \mathbb{R}^d$. In the binary case, the Hamming distance is reduced to the 0-1 error function, $H(y, \hat{y}_{\boldsymbol{w}}) = \mathbf{1}[y \neq \hat{y}_{\boldsymbol{w}}]$, therefore

$$\max_{\hat{\boldsymbol{y}}\in\mathcal{Y}}\left[H(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}) - \boldsymbol{w}^\top\boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}) + \boldsymbol{w}^\top\boldsymbol{\phi}(\boldsymbol{x}, \hat{\boldsymbol{y}})\right] = \max\{0, 1-y\,\boldsymbol{w}^\top\boldsymbol{\psi}(\boldsymbol{x})\}, \tag{1.8}$$

which is the binary hinge loss function.

We can also justify the use of the Hamming hinge loss as a surrogate loss function from generalization bounds, which were given by Taskar et al. (2003) and by McAllester (2006). In particular, consider the following gen-

1. Personal communication with David McAllester.

eralization theorem stated in (McAllester, 2006, Theorem 62):

**Theorem 1.1.** *Assume that $0 \le \ell(\boldsymbol{y}, \hat{\boldsymbol{y}}) \le 1$. With probability at least $1 - \delta$ over the draw of the training set $\mathcal{S}$ of size $m$, the following holds simultaneously for all weight vectors $\boldsymbol{w}$*

$$\begin{aligned}\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\rho}&[\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x}))] \\ &\le \frac{1}{m}\sum_{i=1}^{m} \max_{\hat{\boldsymbol{y}}} \mathbf{1}\left[\boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{y}_i) - \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}_i, \hat{\boldsymbol{y}}) \le H(\boldsymbol{y}_i, \hat{\boldsymbol{y}})\right] \ell(\boldsymbol{y}_i, \hat{\boldsymbol{y}}) \\ &\qquad + \frac{\|\boldsymbol{w}\|^2}{m} + \sqrt{\frac{\|\boldsymbol{w}\|^2 \ln\left(\frac{2dm}{\|\boldsymbol{w}\|^2}\right) + \ln\left(\frac{m}{\delta}\right)}{2(m-1)}}, \end{aligned} \tag{1.9}$$

*where $\mathbf{1}[\pi]$ denotes the indicator function: $\mathbf{1}[\pi] = 1$ if the predicate $\pi$ is true and $0$ otherwise.*

This generalization bound provides a rationalization of the use of Hamming hinge loss as a surrogate loss when the conditions $\boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{y}_i) - \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}_i, \hat{\boldsymbol{y}}) \le H(\boldsymbol{y}_i, \hat{\boldsymbol{y}})$ hold for all $1 \le i \le m$. Similar rationalization can be derived from the bounds stated by Taskar et al. (2003).

Another formulation of large margin structured predictors is called *structural SVM* and was introduced by Tsochantaridis et al. (2005) with two variations. The first variation is called *margin-scaled* and is aimed at minimizing the following surrogate loss

$$\bar{\ell}_{margin}(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y}) = \max_{\hat{\boldsymbol{y}} \in \mathcal{Y}} \left[\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}) - \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}) + \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}, \hat{\boldsymbol{y}})\right]. \tag{1.10}$$

The second variation is called *slack-scaled*, and aimed at minimizing the following surrogate loss

$$\bar{\ell}_{slack}(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y}) = \max_{\hat{\boldsymbol{y}} \in \mathcal{Y}} \ \ell(\boldsymbol{y}, \hat{\boldsymbol{y}}) \left[1 - \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}) + \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}, \hat{\boldsymbol{y}})\right] \tag{1.11}$$

Both of those surrogate loss functions are justified as being convex upper bounds on the cost function. For the margin-scaled hinge loss we have

$$\begin{aligned} \ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}) &= \ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}) - \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}) + \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}) && (1.12) \\ &\le \ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}) - \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}) + \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}) && (1.13) \\ &\le \max_{\hat{\boldsymbol{y}} \in \mathcal{Y}} \Big(\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}) - \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}) + \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}, \hat{\boldsymbol{y}})\Big) && (1.14) \\ &\le \bar{\ell}_{margin}(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y}) && (1.15) \end{aligned}$$

where we used the definition of $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$ in (1.1) for bounding (1.12) by (1.13). Likewise, a similar upper-bound can be derived for the slack-scaled hinge loss. The upper bound convert the optimization problem (1.3) to a convex problem, which is of great technical and theoretical advantage. However, we are unaware of a guarantee that ensures that minimizing the convex upper-bound $\bar{\ell}_{margin}$ or $\bar{\ell}_{slack}$ also minimizes the cost $\ell$. This type of guarantees are often given in the analysis of approximation algorithms in terms of a multiplicative factor times the minimal objective, but we are unaware of any such result for the hinge loss.

The weight vector $\boldsymbol{w}$ can be found by several optimization techniques. The use of SSGD to solve (1.3) will help us to compare the update rule of $\boldsymbol{w}$ of different methods. On iteration $t$ we choose a random training example $(\boldsymbol{x}_{j_t}, \boldsymbol{y}_{j_t})$ by picking an index $j_t \in \{1, \ldots, m\}$ uniformly at random. Then we replace the objective in (1.3) where the surrogate loss is $\bar{\ell}_{margin}$ with an approximation based on the training example $(\boldsymbol{x}_{j_t}, \boldsymbol{y}_{j_t})$. The sub-gradient of approximated objective is given by

$$\nabla_t = \boldsymbol{\phi}(\boldsymbol{x}_{j_t}, \hat{\boldsymbol{y}}^{\ell}_{\boldsymbol{w}_t}) - \boldsymbol{\phi}(\boldsymbol{x}_{j_t}, \boldsymbol{y}_{j_t}) + \lambda \boldsymbol{w}_t \tag{1.16}$$

where

$$\hat{\boldsymbol{y}}^{\ell}_{\boldsymbol{w}_t} = \underset{\boldsymbol{y} \in \mathcal{Y}}{\mathrm{argmax}} \; \boldsymbol{w}_t^\top \boldsymbol{\phi}(\boldsymbol{x}_{j_t}, \boldsymbol{y}) + \ell(\boldsymbol{y}_{j_t}, \boldsymbol{y}) \tag{1.17}$$

We call this type of inference *loss augmented inference*. Hence the we have following update rule:

$$\boldsymbol{w}_{t+1} = (1 - \eta_t \lambda)\, \boldsymbol{w}_t + \eta_t \left( \boldsymbol{\phi}(\boldsymbol{x}_{j_t}, \boldsymbol{y}_{j_t}) - \boldsymbol{\phi}(\boldsymbol{x}_{j_t}, \hat{\boldsymbol{y}}^{\ell}_{\boldsymbol{w}_t}) \right) \tag{1.18}$$

There are two main differences between the large margin algorithms and the "batch" Perceptron algorithm, namely, the regularization and the margin, Those differences can be seen by comparing the SVM update rule in (1.18) to the Perceptron update in (1.5). Note that similar derivation can be used to find the update rule of the slack-scaled loss function and of the Hamming loss function.

We would like to finish this section with a short discussion on consistency of the hinge loss functions. The large margin structured predictors are not consistent in the sense of (1.4), that is, they fail to converge on the optimal linear predictor even in the limit of infinite training data. It was shown (Lee et al., 2004) that the use of hinge loss in multiclass classification results in inconsistent algorithms. This claim can be extended to prove that the structural hinge losses are also inconsistent (McAllester, 2006).

## 1.4 Conditional Random Fields

Conditional random fields (CRFs) that were proposed by (Lafferty et al., 2001) are models which use the negative log-likelihood loss ("log loss") function as a surrogate loss function in (1.3), that is

$$\bar{\ell}_{log}(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y}) = -\ln P_{\boldsymbol{w}}(\boldsymbol{y} \,|\, \boldsymbol{x}) \tag{1.19}$$

where

$$P_{\boldsymbol{w}}(\boldsymbol{y} \,|\, \boldsymbol{x}) = \frac{1}{Z_{\boldsymbol{w}}(\boldsymbol{x})} \exp\left\{\boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y})\right\} \tag{1.20}$$

and $Z_{\boldsymbol{w}}(\boldsymbol{x})$ is the partition function defined as

$$Z_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_{\boldsymbol{y}' \in \mathcal{Y}} \exp\left\{\boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}')\right\}. \tag{1.21}$$

This defines a convex optimization problem. With SSGD the update rule for an example $j_t$ is

$$\boldsymbol{w}_{t+1} = (1 - \eta_t \lambda)\, \boldsymbol{w}_t + \eta_t \Big( \boldsymbol{\phi}(\boldsymbol{x}_{j_t}, \boldsymbol{y}_{j_t}) - \mathbb{E}_{\boldsymbol{y}' \sim P_w(\boldsymbol{y}' \,|\, \boldsymbol{x})} \left[\boldsymbol{\phi}(\boldsymbol{x}_{j_t}, \boldsymbol{y}')\right] \Big). \tag{1.22}$$

where

$$\mathbb{E}_{\boldsymbol{y}' \sim P_w(\boldsymbol{y}' \,|\, \boldsymbol{x})} \left[\boldsymbol{\phi}(\boldsymbol{x}_{j_t}, \boldsymbol{y}')\right] = \sum_{\boldsymbol{y}' \in \mathcal{Y}} P_{\boldsymbol{w}}(\boldsymbol{y}' \,|\, \boldsymbol{x})\, \boldsymbol{\phi}(\boldsymbol{x}_{j_t}, \boldsymbol{y}'). \tag{1.23}$$

Given a conditional probability model $P_{\boldsymbol{w}}(\boldsymbol{y} \,|\, \boldsymbol{x})$, when $\boldsymbol{w}$ is found according to (1.22), one could use decision-theoretic prediction defined as follows

$$\hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x}) = \underset{\hat{\boldsymbol{y}}}{\operatorname{argmin}}\ \mathbb{E}_{\boldsymbol{y} \sim P_w(\boldsymbol{y} \,|\, \boldsymbol{x})} \left[\ell(\boldsymbol{y}, \hat{\boldsymbol{y}})\right] \tag{1.24}$$

If $P_{\boldsymbol{w}}(\boldsymbol{y} \,|\, \boldsymbol{x})$ equals $\rho(\boldsymbol{y}|\boldsymbol{x})$ then (1.24) gives an optimal decoding. For example, if the cost function is the zero-one loss, $\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}) = \mathbf{1}[\boldsymbol{y} \neq \hat{\boldsymbol{y}}_{\boldsymbol{w}}]$, then from (1.24) we get the Bayes optimal decoder

$$P_{\boldsymbol{w}}(\boldsymbol{y} \,|\, \boldsymbol{x}) = \underset{\boldsymbol{y}'}{\operatorname{argmin}}\, \mathbb{E}_{\boldsymbol{y} \sim P_w(\boldsymbol{y} \,|\, \boldsymbol{x})} \left[\mathbf{1}[\boldsymbol{y} \neq \boldsymbol{y}']\right] = \underset{\boldsymbol{y}'}{\operatorname{argmax}}\, P(\boldsymbol{y} = \boldsymbol{y}'|\boldsymbol{x}). \tag{1.25}$$

However, since the log loss function (1.19) is defined independently of the cost function $\ell$, the optimum of (1.3) with $\bar{\ell}_{log}$ is not expected to yield good solutions to (1.2).

## 1.5 Direct Loss Minimization

Direct loss minimization is a method focused at directly optimizing the measure of performance. As we have seen, the most common approaches to structured prediction, structural Perception, M$^3$, structural SVMs and CRFs, do not minimize the risk directly. Note that except the structured Perceptron, all those approaches minimize a convex regularized surrogate loss function. An alternative to a convex relaxation is to perform SSGD directly on the objective in (1.2). This is conceptually puzzling in the case where the output space $\mathcal{Y}$ is discrete since the output $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$ is not a differentiable function of $\boldsymbol{w}$. In that case the gradient of the risk $\nabla_{\boldsymbol{w}}\mathbb{E}[\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x}))]$ does not equal to the expected gradient of the cost $\mathbb{E}[\nabla_{\boldsymbol{w}}\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x}))]$. However, McAllester et al. (2010) showed that when the input space $\mathcal{X}$ is continuous the gradient $\nabla_{\boldsymbol{w}}\mathbb{E}[\ell(\boldsymbol{y}, \boldsymbol{y}_{\boldsymbol{w}}(\boldsymbol{x}))]$ exists even when the output space $\mathcal{Y}$ is discrete.

**Theorem 1.2.** *For a finite set $\mathcal{Y}$ of possible output values, and for $w$ in general position as defined in McAllester et al. (2010), we have the following*

$$\nabla_{\boldsymbol{w}}\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\rho}\Big[\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x}))\Big] = \lim_{\epsilon\to 0} \frac{\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\rho}\left[\boldsymbol{\phi}(\boldsymbol{x}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}^{\epsilon\ell}(\boldsymbol{x})) - \boldsymbol{\phi}(\boldsymbol{x}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x}))\right]}{\epsilon} \tag{1.26}$$

*where*

$$\hat{\boldsymbol{y}}_{\boldsymbol{w}}^{\epsilon\ell}(\boldsymbol{x}) = \operatorname*{argmax}_{\hat{\boldsymbol{y}}\in\mathcal{Y}} \; \boldsymbol{w}^\top\boldsymbol{\phi}(\boldsymbol{x}, \hat{\boldsymbol{y}}) + \epsilon\,\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}) \tag{1.27}$$

*and $\hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x})$ is defined as in (1.1).*

Consider the definition of the gradient of the risk using Frèchet derivative

$$\Delta\boldsymbol{w}^\top\nabla_{\boldsymbol{w}}\mathbb{E}[\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x}))] = \lim_{\epsilon\to 0} \frac{\mathbb{E}\left[\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}+\epsilon\Delta\boldsymbol{w}}(\boldsymbol{x})) - \ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x}))\right]}{\epsilon}. \tag{1.28}$$

for any unit vector $\Delta\boldsymbol{w} \in \mathbb{R}^d$. The main idea of the proof is to show that the right hand side of (1.28) equals the right hand side of (1.26). This can be shown as follows. Express the expectation of the right hand side of (1.26) as a surface integral over the decision boundary, when this boundary is with respect to the score of switching the label from $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$ to $\hat{\boldsymbol{y}}_{\boldsymbol{w}}^{\epsilon\ell}$. Similarly, express the expectation of the right hand side of (1.28) as a surface integral over the decision boundary, when switching the label from $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$ to $\hat{\boldsymbol{y}}_{\boldsymbol{w}+\epsilon\Delta\boldsymbol{w}}$. Then, compare those two integrals analytically. The proof of the theorem is given in (McAllester et al., 2010).

Using SSGD, an update rule can be written as follows

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \frac{\eta_t}{\epsilon}\left(\boldsymbol{\phi}(\boldsymbol{x}_{j_t}, \hat{\boldsymbol{y}}_{\boldsymbol{w}_t}(\boldsymbol{x}_{j_t})) - \boldsymbol{\phi}(\boldsymbol{x}_{j_t}, \hat{\boldsymbol{y}}^{\epsilon\ell}_{\boldsymbol{w}_t}(\boldsymbol{x}_{j_t}))\right). \tag{1.29}$$

This type of update rule *moves away from worse labels*. In similar fashion, we can derive an update rule which *moves toward better labels* and usually performs better in practice:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \frac{\eta_t}{\epsilon}\left(\boldsymbol{\phi}(\boldsymbol{x}_{j_t}, \hat{\boldsymbol{y}}^{-\epsilon\ell}_{\boldsymbol{w}_t}(\boldsymbol{x}_{j_t})) - \boldsymbol{\phi}(\boldsymbol{x}_{j_t}, \hat{\boldsymbol{y}}_{\boldsymbol{w}_t}(\boldsymbol{x}_{j_t}))\right). \tag{1.30}$$

This update was obtained by expressing the gradient in (1.28) with labels $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$ and $\hat{\boldsymbol{y}}_{\boldsymbol{w}-\epsilon\Delta\boldsymbol{w}}$ rather than $\hat{\boldsymbol{y}}_{\boldsymbol{w}+\epsilon\Delta\boldsymbol{w}}$ and $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$. In practice, $\epsilon$ can be chosen to be fixed on a held-out development set.

An open problem is how to properly incorporate regularization in the case where only a finite number of training example is available. It should be noted that naive regularization with a norm of $\boldsymbol{w}$, such as regularizing with $\lambda\|\boldsymbol{w}\|^2$, is nonsensical as the risk $\mathbb{E}[\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x}))]$ is insensitive to the norm of $\boldsymbol{w}$. Early stopping may be a viable approach in practice.

## 1.6 Structured Ramp Loss

One approach to add a regularization term to the direct loss minimization update rule was proposed by McAllester and Keshet (2011). The idea is to use *structured ramp loss* (Do et al., 2008) as a surrogate loss function, which is defined as follows:

$$\bar{\ell}_{ramp}(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y}) = \max_{\hat{\boldsymbol{y}}\in\mathcal{Y}}\left[\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}) + \boldsymbol{w}^\top\boldsymbol{\phi}(\boldsymbol{x}, \hat{\boldsymbol{y}})\right] - \max_{\tilde{\boldsymbol{y}}\in\mathcal{Y}}\left[\boldsymbol{w}^\top\boldsymbol{\phi}(\boldsymbol{x}, \tilde{\boldsymbol{y}})\right]. \tag{1.31}$$

Using this surrogate loss in the optimization problem (1.3) results in a non-convex optimization problem. Finding the sub-gradient of (1.3) with $\bar{\ell}_{ramp}$ and applying SSGD, we get the following update rule:

$$\boldsymbol{w}_{t+1} = (1 - \eta_t\lambda)\,\boldsymbol{w}_t + \eta_t\left(\boldsymbol{\phi}(\boldsymbol{x}_{j_t}, \hat{\boldsymbol{y}}_{\boldsymbol{w}_t}(\boldsymbol{x}_{j_t})) - \boldsymbol{\phi}(\boldsymbol{x}_{j_t}, \hat{\boldsymbol{y}}^{\ell}_{\boldsymbol{w}_t}(\boldsymbol{x}_{j_t}))\right). \tag{1.32}$$

This update is similar to the direct loss minimization update rule (1.29), except for the missing $\epsilon$.

Conceptually, it seems that $\|\boldsymbol{w}\|$ in (1.32) serves as $1/\epsilon$ in the direct loss update rule: high value of $\|\boldsymbol{w}\|$ has the same effect as small $\epsilon$ in (1.29). According to Theorem 1.2 the sub-gradient of the risk equals the expected difference in the feature maps when $\epsilon$ approaches zero. For the ramp loss, on the other hand, it can be shown (McAllester and Keshet, 2011) that when the norm of $\boldsymbol{w}$ goes to infinity, then the ramp loss approaches the risk.

The regularization of the structural ramp loss $\lambda\|\boldsymbol{w}\|^2$ prefer weight vectors $\boldsymbol{w}$ with small norms. Those conditions are somewhat contradicting, and it is an open problem to introduce a better regularization term with the structured ramp loss.

The structured ramp loss is consistent in the sense that was defined in (1.4). No convex surrogate loss function, such as log loss or hinge loss, can be consistent in this sense – for any nontrivial convex surrogate loss function one can give examples (a single feature suffices) where the learned weight vector is perturbed by outliers but where the outliers do not actually influence the optimal cost.

The structured ramp loss in (1.31) corresponds to the "away-from-bad" direct loss update version. Consider the following variant to the structured ramp loss:

$$\bar{\ell}'_{ramp}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{y}) = \max_{\tilde{\boldsymbol{y}}\in\mathcal{Y}}\left[\boldsymbol{w}^\top\boldsymbol{\phi}(\boldsymbol{x},\tilde{\boldsymbol{y}})\right] - \max_{\hat{\boldsymbol{y}}\in\mathcal{Y}}\left[\boldsymbol{w}^\top\boldsymbol{\phi}(\boldsymbol{x},\hat{\boldsymbol{y}}) - \ell(\boldsymbol{y},\hat{\boldsymbol{y}})\right]. \tag{1.33}$$

The sub-gradient update equation for $\bar{\ell}'_{ramp}$ defines an update rule corresponds to the "toward good" version of the direct loss minimization. However, we are unaware of a method for proving consistency for this surrogate loss function, and the method used in McAllester and Keshet (2011) is not suitable for this case.

## 1.7 Structured Probit Loss

We turn now to describe a different approach which is based on the concept of perturbations of the weight vector $\boldsymbol{w}$ (Keshet et al., 2011). Define the structured probit loss as follows:

$$\bar{\ell}_{probit}(\boldsymbol{w},\boldsymbol{x},\boldsymbol{y}) = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I})}\left[\ell(\boldsymbol{y},\hat{\boldsymbol{y}}_{\boldsymbol{w}+\boldsymbol{\epsilon}}(\boldsymbol{x}))\right]. \tag{1.34}$$

where $\boldsymbol{\epsilon}\in\mathbb{R}^d$ is a random vector drawn from the isotropic normal distribution. Note that the optimization problem (1.3) where the surrogate loss is the structured probit loss is a non-convex optimization function. Plugging this loss into (1.3) we have:

$$\boldsymbol{w}^* = \operatorname*{argmin}_{\boldsymbol{w}} \frac{1}{m}\sum_{i=1}^{m}\bar{\ell}_{probit}(\boldsymbol{w},\boldsymbol{x}_i,\boldsymbol{y}_i) + \frac{\lambda}{2}\|\boldsymbol{w}\|^2. \tag{1.35}$$

The sub-gradient of the objective (1.35), approximated with the training example $(\boldsymbol{x}_{j_t}, \boldsymbol{y}_{j_t})$ uniformly chosen at random, is given by

$$\nabla_{\boldsymbol{w}} \left[ \bar{\ell}_{probit}(\boldsymbol{w}, \boldsymbol{x}_{j_t}, \boldsymbol{y}_{j_t}) + \frac{\lambda}{2}\|\boldsymbol{w}\|^2 \right] \tag{1.36}$$

$$= \nabla_{\boldsymbol{w}} \left[ \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I})} \left[ \ell(\boldsymbol{y}_{j_t}, \hat{\boldsymbol{y}}_{\boldsymbol{w}+\boldsymbol{\epsilon}}(\boldsymbol{x}_{j_t})) \right] + \frac{\lambda}{2}\|\boldsymbol{w}\|^2 \right] \tag{1.37}$$

$$= \nabla_{\boldsymbol{w}} \left[ \int (2\pi)^{-d/2} e^{-\frac{1}{2}\|\boldsymbol{\epsilon}\|^2} \ell(\boldsymbol{y}_{j_t}, \hat{\boldsymbol{y}}_{\boldsymbol{w}+\boldsymbol{\epsilon}}(\boldsymbol{x}_{j_t})) d\boldsymbol{\epsilon} + \frac{\lambda}{2}\|\boldsymbol{w}\|^2 \right] \tag{1.38}$$

$$= \nabla_{\boldsymbol{w}} \left[ (2\pi)^{-d/2} \int e^{-\frac{1}{2}\|\boldsymbol{u}-\boldsymbol{w}\|^2} \ell(\boldsymbol{y}_{j_t}, \hat{\boldsymbol{y}}_{\boldsymbol{u}}(\boldsymbol{x}_{j_t})) d\boldsymbol{u} + \frac{\lambda}{2}\|\boldsymbol{w}\|^2 \right] \tag{1.39}$$

$$= (2\pi)^{-d/2} \int (\boldsymbol{u}-\boldsymbol{w})\, e^{-\frac{1}{2}\|\boldsymbol{u}-\boldsymbol{w}\|^2} \ell(\boldsymbol{y}_{j_t}, \hat{\boldsymbol{y}}_{\boldsymbol{u}}(\boldsymbol{x}_{j_t})) d\boldsymbol{u} + \lambda \boldsymbol{w} \tag{1.40}$$

$$= (2\pi)^{-d/2} \int \boldsymbol{\epsilon}\, e^{-\frac{1}{2}\|\boldsymbol{\epsilon}\|^2} \ell(\boldsymbol{y}_{j_t}, \hat{\boldsymbol{y}}_{\boldsymbol{w}+\boldsymbol{\epsilon}}(\boldsymbol{x}_{j_t})) d\boldsymbol{\epsilon} + \lambda \boldsymbol{w} \tag{1.41}$$

$$= \mathbb{E}_{\boldsymbol{\epsilon}} \left[ \boldsymbol{\epsilon}\, \ell(\boldsymbol{y}_{j_t}, \hat{\boldsymbol{y}}_{\boldsymbol{w}+\boldsymbol{\epsilon}}(\boldsymbol{x}_{j_t})) \right] + \lambda \boldsymbol{w}. \tag{1.42}$$

where we changed variables $\boldsymbol{\epsilon} = \boldsymbol{u} - \boldsymbol{w}$ in the transition from (1.38) to (1.39) and back to $\boldsymbol{u} = \boldsymbol{w} + \boldsymbol{\epsilon}$ in the transition from (1.40) to (1.41). The update rule is therefore

$$\boldsymbol{w}_{t+1} = (1 - \eta_t \lambda)\, \boldsymbol{w}_t + \eta_t \mathbb{E}_{\boldsymbol{\epsilon}} \left[ \boldsymbol{\epsilon}\, \ell(\boldsymbol{y}_{j_t}, \hat{\boldsymbol{y}}_{\boldsymbol{w}+\boldsymbol{\epsilon}}(\boldsymbol{x}_{j_t})) \right] \tag{1.43}$$

Practically the expectation over the isotropic normal random noise $\boldsymbol{\epsilon}$ is replace with an average of a vector of length $d$ sampled from that distribution.

The following generalization bound was given in (Keshet et al., 2011).

**Theorem 1.3.** *For fixed $\lambda > 1/2$ we have that with probability at least $1-\delta$ over the draw of the training data the following holds simultaneously for all $\boldsymbol{w}$*

$$\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\rho}[\bar{\ell}_{probit}(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y})] \le \frac{1}{1-\frac{1}{2\lambda}} \Big( \frac{1}{m} \sum_{i=1}^{m} \bar{\ell}_{probit}(\boldsymbol{w}, \boldsymbol{x}_i, \boldsymbol{y}_i) + \frac{\lambda}{2m}\|\boldsymbol{w}\|^2 + \frac{\lambda \ln(1/\delta)}{m} \Big). \tag{1.44}$$

It is interesting to note that minimizing the right hand side of this bound with respect to $\boldsymbol{w}$, using SSGD is identical to the derivation of the update rule above.

This loss function is found to be consistent in the strong sense as defined in (1.4) (McAllester and Keshet, 2011). Another advantage of this surrogate loss is that the cost function $\ell$ does not need to be decomposable as in structural SVM, direct loss minimization and structural ramp loss. The decomposability is needed to solve loss-adjusted inference (1.17) using dynamic programming[2]. This, however, comes with a disadvantage: the need to infer $\hat{\boldsymbol{y}}_{\boldsymbol{w}+\boldsymbol{\epsilon}}(\boldsymbol{x})$ many times in order to reliably estimate the expectation over $\boldsymbol{\epsilon}$.

An extension of the probit loss to more general distributions of $\boldsymbol{\epsilon}$ was presented by Hazan et al. (2013), which might correspond to the noise of the problem. In that case the regularization term in (1.3) is not necessarily $\|\boldsymbol{w}\|^2$, but rather a function related to the distribution of $\boldsymbol{\epsilon}$.

## 1.8 Risk Minimization under Gibbs distribution

While CRFs aim at minimizing the expected negative log likelihood, namely the log-loss, recall that we are interested in minimizing the risk. It was proposed by Smith and Eisner (2006) to minimize the risk under the Gibbs measure. Let us define the *structured logit* surrogate loss function as the conditional expectation of the cost, when the expectation is taken with respect to the *Gibbs distribution*, $P_{\boldsymbol{w}}(y|x)$, as is defined in (1.19):

$$\bar{\ell}_{logit}(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y}) = \mathbb{E}_{\boldsymbol{y}' \sim P_{\boldsymbol{w}}(\boldsymbol{y}' \,|\, \boldsymbol{x})} \left[ \ell(\boldsymbol{y}', \boldsymbol{y}) \right] = \sum_{\boldsymbol{y}' \in \mathcal{Y}} P_{\boldsymbol{w}}(\boldsymbol{y}'|\boldsymbol{x}) \, \ell(\boldsymbol{y}', \boldsymbol{y}). \quad (1.45)$$

An interesting property of the logit loss $\ell_{logit}$ is that when the norm of $\boldsymbol{w}$ approaches infinity then the logit loss converges to the risk.

**Theorem 1.4.**

$$\lim_{\alpha \to \infty} \bar{\ell}_{logit}(\alpha \boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y}) = \ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x})), \quad (1.46)$$

*where $\hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x})$ is defined in (1.1).*

This theorem can be easily proven by explicitly expressing the logit loss as in (1.45). Then split the sum over the labels to the label $\hat{\boldsymbol{y}}$ and the rest of the labels, and apply the limit.

This theorem is a only part of whole consistency prove. What left is a generalization bound similar to (1.44), that makes the connection between the logit loss and the risk. This is still an open problem.

2. In principle, the decomposability is not always necessary in those methods, as long as the loss augmented prediction still works, e.g. by branch-and-bound (Blaschko and Lampert, 2008)

## 1.9 Conclusions

In this chapter we compared different surrogate loss functions used by different algorithms for structured prediction. We presented the concept of consistency in the strong sense, and showed that no convex surrogate loss function, such as log loss or hinge loss, can be consistent in this sense. We showed that some non-convex loss functions lead to consistency, and maybe superior generalization. We have started to extend the ideas presented here to train models, such as graphical models and deep neural networks, so as to optimize the measure of performance on unseen data.

## 1.10 References

M. B. Blaschko and C. H. Lampert. Learning to localize objects with structured output regression. In *European Conference on Computer Vision (ECCV)*, pages 2–15. Springer, 2008.

N. Cesa-Bianchi, A. Conconi, and C. Gentile. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory*, 50(9): 2050–2057, 2004.

M. Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Conference on Empirical Methods in Natural Language Processing*, 2002.

O. Dekel, J. Keshet, and Y. Singer. Large margin hierarchical classification. In *Proceedings of the twenty-first international conference on Machine learning (ICML)*, 2004.

C. B. Do, Q. Le, C. H. Teo, O. Chapelle, and A. Smola. Tighter bounds for structured estimation. In *Advances in neural information processing systems (NIPS) 22*, pages 281–288, 2008.

T. Hazan, S. Maji, J. Keshet, and T. Jaakkola. On sampling from the Gibbs distribution with random maximum a-posteriori perturbations. In *Neural Information and Processing Systems (NIPS) 27*, 2013.

J. Keshet, D. McAllester, and T. Hazan. PAC-Bayesian approach for minimization of phoneme error rate. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2011.

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 (ICML)*, pages 282–289, 2001.

Y. Lee, Y. Lin, and G. Wahba. Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99(465):67–81, 2004.

D. McAllester. Generalization bounds and consistency for structured labeling. In B. Schölkopf, A. J. Smola, B. Taskar, and S. Vishwanathan, editors, *Predicting Structured Data*, pages 247–262. MIT Press, 2006.

D. McAllester and J. Keshet. Generalization bounds and consistency for latent structural probit and ramp loss. In *Advances in Neural Information Processing Systems (NIPS) 25*, 2011.

D. McAllester, T. Hazan, and J. Keshet. Direct loss minimization for structured prediction. In *Advances in Neural Information Processing Systems (NIPS) 24*, 2010.

D. A. Smith and J. Eisner. Minimum risk annealing for training log-linear models. In *Proc. of the COLING/ACL*, pages 787–794, 2006.

B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *Advances in Neural Information Processing Systems 17*, 2003.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

V. Vapnik. *The nature of statistical learning theory.* springer, 2000.