Aggressive Online Learning of Structured Classifiers

Andre F. T. Martins[†‡] **Kevin Gimpel**[†]
Noah A. Smith[†] **Eric P. Xing**[†]
Mario A. T. Figueiredo[‡] **Pedro M. Q. Aguiar**[♯]

June 2010
CMU-ML-08-106

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

[†]School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA,
[‡]Instituto de Telecomunicações / [♯]Instituto de Sistemas e Robótica, Instituto Superior Técnico, Lisboa, Portugal

A. M. was supported by a grant from FCT/ICTI through the CMU-Portugal Program, and also by Priberam Informática. N. S. was supported in part by Qatar NRF NPRP-08-485-1-083. E. X. was supported by AFOSR FA9550010247, ONR N000140910758, NSF CAREER DBI-0546594, NSF IIS-0713379, and an Alfred P. Sloan Fellowship. M. F. and P. A. were supported by the FET programme (EU FP7), under the SIMBAD project (contract 213250).

Keywords: Structured prediction, online learning, dual coordinate ascent

Abstract

We present a unified framework for online learning of structured classifiers that handles a wide family of convex loss functions, properly including CRFs, structured SVMs, and the structured perceptron. We introduce a new aggressive online algorithm that optimizes any loss in this family. For the structured hinge loss, this algorithm reduces to 1-best MIRA; in general, it can be regarded as a dual coordinate ascent algorithm. The approximate inference scenario is also addressed. Our experiments on two NLP problems show that the algorithm converges to accurate models at least as fast as stochastic gradient descent, without the need to specify any learning rate parameter.

1 Introduction

Learning structured classifiers discriminatively typically involves the minimization of a regularized loss function; the well-known cases of conditional random fields (CRFs, [Lafferty et al., 2001]) and structured support vector machines (SVMs, [Taskar et al., 2003, Tsochantaridis et al., 2004, Altun et al., 2003]) correspond to different choices of loss functions. For large-scale settings, the underlying optimization problem is often difficult to tackle in its batch form, increasing the popularity of online algorithms. Examples are the structured perceptron [Collins, 2002a], stochastic gradient descent (SGD) [LeCun et al., 1998], and the margin infused relaxed algorithm (MIRA) [Crammer et al., 2006].

This paper presents a unified representation for several convex loss functions of interest in structured classification (§2). In §3, we describe how all these losses can be expressed in variational form as optimization problems over the marginal polytope [Wainwright and Jordan, 2008]. We make use of convex duality to derive new online learning algorithms (§4) that share the "passive-aggressive" property of MIRA but can be applied to a wider variety of loss functions, including the logistic loss that underlies CRFs. We show that these algorithms implicitly perform coordinate ascent in the dual, generalizing the framework in Shalev-Shwartz and Singer [2006] for a larger set of loss functions and for structured outputs.

The updates we derive in §4 share the remarkable simplicity of SGD, with an important advantage: they do not require tuning a learning rate parameter or specifying an annealing schedule. Instead, the step sizes are a function of the loss and its gradient. The additional computation required for loss evaluations is negligible since the methods used to compute the gradient also provide the loss value.

Two important problems in NLP provide an experimental testbed (§5): named entity recognition and dependency parsing. We employ feature-rich models where exact inference is sometimes intractable. To be as general as possible, we devise a framework that fits any structured classification problem representable as a factor graph with soft and hard constraints (§2); this includes problems with loopy graphs, such as some variants of the dependency parsers of Smith and Eisner [2008].

2 Structured Classification and Loss Functions

2.1 Inference and Learning

Denote by $\mathcal{X}$ a set of input objects from which we want to infer some hidden structure conveyed in an output set $\mathcal{Y}$. We assume a supervised setting, where we are given labeled data $\mathcal{D} \triangleq \{(x_1, y_1), \ldots, (x_m, y_m)\} \subseteq \mathcal{X} \times \mathcal{Y}$. Each input $x \in \mathcal{X}$ (*e.g.*, a sentence) is associated with a set of legal outputs $\mathcal{Y}(x) \subseteq \mathcal{Y}$ (*e.g.*, candidate parse trees); we are interested in the case where $\mathcal{Y}(x)$ is a structured set whose cardinality grows exponentially with the size of x. We consider linear classifiers $h_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$ of the form

$$h_{\boldsymbol{\theta}}(x) \triangleq \operatorname*{argmax}_{y \in \mathcal{Y}(x)} \boldsymbol{\theta}^\top \boldsymbol{\phi}(x, y), \tag{1}$$

where $\boldsymbol{\theta} \in \mathbb{R}^d$ is a vector of parameters and $\boldsymbol{\phi}(x, y) \in \mathbb{R}^d$ is a feature vector. Our goal is to learn the parameters $\boldsymbol{\theta}$ from the data $\mathcal{D}$ such that $h_{\boldsymbol{\theta}}$ has small generalization error. We assume a cost function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ is given, where $\ell(\hat{y}, y)$ is the cost of predicting $\hat{y}$ when the true output is y. Typically, direct minimization of the empirical risk, $\min_{\boldsymbol{\theta} \in \mathbb{R}^d} \frac{1}{m} \sum_{i=1}^m \ell(h_{\boldsymbol{\theta}}(x_i), y_i)$, is intractable and hence a surrogate non-negative, convex loss $L(\boldsymbol{\theta}; x, y)$ is used. To avoid overfitting, a regularizer $R(\boldsymbol{\theta})$ is added, yielding the learning problem

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^d} \lambda R(\boldsymbol{\theta}) + \frac{1}{m} \sum_{i=1}^m L(\boldsymbol{\theta}; x_i, y_i), \tag{2}$$

where $\lambda \in \mathbb{R}$ is the regularization coefficient. Throughout this paper we assume ℓ_2-regularization, $R(\boldsymbol{\theta}) \triangleq \frac{1}{2}\|\boldsymbol{\theta}\|^2$, and focus on loss functions of the form

$$L_{\beta,\gamma}(\boldsymbol{\theta}; x, y) \triangleq \frac{1}{\beta} \log \sum_{y' \in \mathcal{Y}(x)} \exp\left[\beta\Big(\boldsymbol{\theta}^\top\big(\boldsymbol{\phi}(x, y') - \boldsymbol{\phi}(x, y)\big) + \gamma \ell(y', y)\Big)\right], \tag{3}$$

which subsumes some well-known cases:

- The *logistic loss* (in CRFs), $L_{\text{CRF}}(\boldsymbol{\theta}; x, y) \triangleq -\log P_{\boldsymbol{\theta}}(y|x)$, corresponds to $\beta = 1$ and $\gamma = 0$;
- The *hinge loss* of structured SVMs, $L_{\text{SVM}}(\boldsymbol{\theta}; x, y) \triangleq \max_{y' \in \mathcal{Y}(x)} \boldsymbol{\theta}^\top(\boldsymbol{\phi}(x, y') - \boldsymbol{\phi}(x, y)) + \ell(y', y)$, corresponds to the limit case $\beta \to \infty$ and any $\gamma > 0$;
- The loss underlying the structured perceptron is obtained for $\beta \to \infty$ and $\gamma = 0$.
- The *softmax-margin loss* recently proposed in Gimpel and Smith [2010] is obtained with $\beta = \gamma = 1$.

For any choice of $\beta > 0$ and $\gamma \geq 0$, the resulting loss function is convex in $\boldsymbol{\theta}$, since, up to a scale factor, it is the composition of the (convex) log-sum-exp function with an affine map.[1] In §4 we present a dual coordinate ascent online algorithm to handle (2), for this family of losses.

2.2 A Framework for Structured Inference

Two important inference problems are: to obtain the most probable assignment (*i.e.*, to solve (1)) and to compute marginals, when a distribution is defined on $\mathcal{Y}(x)$. Both problems can be challenging when the output set is structured. Typically, there is a natural representation of the elements of $\mathcal{Y}(x)$ as discrete-valued vectors $y \equiv \mathbf{y} = (y_1, \ldots, y_I) \in \mathcal{Y}_1 \times \ldots \times \mathcal{Y}_I \equiv \bar{\mathcal{Y}}$, each $\mathcal{Y}_i$ being a set of labels (I may depend on x). We consider subsets $S \subseteq \{1, \ldots, I\}$ and write partial assignment vectors as $\mathbf{y}_S = (y_i)_{i \in S}$. We assume a one-to-one map (not necessarily onto) from $\mathcal{Y}(x)$ to $\bar{\mathcal{Y}}$ and denote by $\mathcal{S}(x) \subseteq \bar{\mathcal{Y}}$ the subset of representations that correspond to valid outputs.

The next step is to design how the feature vector $\boldsymbol{\phi}(x, y)$ decomposes, which can be conveniently done via a *factor graph* [Kschischang et al., 2001, McCallum et al., 2009]. This is a

[1] Some important non-convex losses can also be written as differences of losses in this family. By defining $\delta L_{\beta,\gamma} = L_{\beta,\gamma} - L_{\beta,0}$, the case $\beta = 1$ yields $\delta L_{\beta,\gamma}(\boldsymbol{\theta}; x, y) = \log \mathbb{E}_{\boldsymbol{\theta}} \exp \ell(Y, y)$, which is an upper bound on $\mathbb{E}_{\boldsymbol{\theta}} \ell(Y, y)$, used in minimum risk training [Smith and Eisner, 2006]. For $\beta = \infty$, $\delta L_{\beta,\gamma}$ becomes a structured *ramp loss* [Collobert et al., 2006].

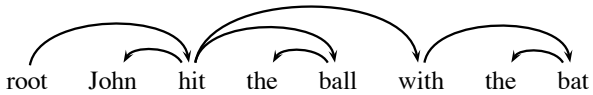

Figure 1: Example of a dependency parse tree (adapted from [McDonald et al., 2005]).

bipartite graph with two types of nodes: variable nodes, which in our case are the I components of $\mathbf{y}$; and a set $\mathcal{C}$ of factor nodes. Each factor node is associated with a subset $C \subseteq \{1, \ldots, I\}$; an edge connects the ith variable node and a factor node C iff $i \in C$. Each factor has a *potential* Ψ_C, a function that maps assignments of variables to non-negative real values. We distinguish between two kinds of factors: *hard constraint factors*, which are used to rule out forbidden partial assignments by mapping them to zero potential values, and *soft factors*, whose potentials are strictly positive. Thus, $\mathcal{C} = \mathcal{C}_{\text{hard}} \cup \mathcal{C}_{\text{soft}}$. We associate with each soft factor a local feature vector $\boldsymbol{\phi}_C(x, \mathbf{y}_C)$ and define

$$\boldsymbol{\phi}(x, y) \triangleq \sum_{C \in \mathcal{C}_{\text{soft}}} \boldsymbol{\phi}_C(x, \mathbf{y}_C). \tag{4}$$

The potential of a soft factor is defined as $\Psi_C(x, \mathbf{y}_C) = \exp(\boldsymbol{\theta}^\top \boldsymbol{\phi}_C(x, \mathbf{y}_C))$. In a log-linear probabilistic model, the feature decomposition in (4) induces the following factorization for the conditional distribution of Y:

$$P_{\boldsymbol{\theta}}(Y = y \mid X = x) = \frac{1}{Z(\boldsymbol{\theta}, x)} \prod_{C \in \mathcal{C}} \Psi_C(x, \mathbf{y}_C), \tag{5}$$

where $Z(\boldsymbol{\theta}, x) = \sum_{\mathbf{y}' \in \mathcal{S}(x)} \prod_{C \in \mathcal{C}} \Psi_C(x, \mathbf{y}'_C)$ is the *partition function*. Two examples follow.

Sequence labeling: Each $i \in \{1, \ldots, I\}$ is a position in the sequence and $\mathcal{Y}_i$ is the set of possible labels at that position. If all label sequences are allowed, then no hard constraint factors exist. In a bigram model, the soft factors are of the form $C = \{i, i+1\}$. To obtain a k-gram model, redefine each $\mathcal{Y}_i$ to be the set of all contiguous $(k-1)$-tuples of labels.

Dependency parsing: In this parsing formalism [Kübler et al., 2009], each input is a sentence (i.e., a sequence of words), and the outputs to be predicted are the dependency arcs, which link *heads* to *modifiers*, and overall must define a spanning tree (see Fig. 1 for an example). We let each $i = (h, m)$ index a pair of words, and define $\mathcal{Y}_i = \{0, 1\}$, where 1 means that there is a link from h to m, and 0 means otherwise. There is one hard factor connected to all variables (call it `TREE`), its potential being one if the arc configurations form a spanning tree and zero otherwise. In the arc-factored model [Eisner, 1996, McDonald et al., 2005], all soft factors are unary and the graph is a tree. More sophisticated models (*e.g.*, with siblings and grandparents) include pairwise factors, creating loops [Smith and Eisner, 2008].

3 Variational Inference

3.1 Polytopes and Duality

Let $\mathcal{P} = \{P_{\boldsymbol{\theta}}(.|x) \mid \boldsymbol{\theta} \in \mathbb{R}^d\}$ be the family of all distributions of the form (5), and rewrite (4) as:

$$\boldsymbol{\phi}(x, y) = \sum_{C \in \mathcal{C}_{\text{soft}}} \boldsymbol{\phi}_C(x, \mathbf{y}_C) = \mathbf{F}(x) \cdot \chi(y),$$

where $\mathbf{F}(x)$ is a d-by-k feature matrix, with $k = \sum_{C \in \mathcal{C}_{\text{soft}}} \prod_{i \in C} |\mathcal{Y}_i|$, each column containing the vectors $\boldsymbol{\phi}_C(x, \mathbf{y}_C)$ for each factor C and configuration $\mathbf{y}_C$; and $\chi(y)$ is a binary k-vector indicating which configurations are active given $Y = y$. We then define the *marginal polytope*

$$\mathcal{Z}(x) \triangleq \text{conv}\{\mathbf{z} \in \mathbb{R}^k \mid \exists y \in \mathcal{Y}(x) \text{ s.t. } \mathbf{z} = \chi(y)\},$$

where conv denotes the convex hull. Note that $\mathcal{Z}(x)$ only depends on the graph and on the specification of the hard constraints (*i.e.*, it is independent of the parameters $\boldsymbol{\theta}$).[2] The next proposition (illustrated in Fig. 2) goes farther by linking the points of $\mathcal{Z}(x)$ to the distributions in $\mathcal{P}$. Below, $H(P_{\boldsymbol{\theta}}(.|x)) = -\sum_{y \in \mathcal{Y}(x)} P_{\boldsymbol{\theta}}(y|x) \log P_{\boldsymbol{\theta}}(y|x)$ denotes the *entropy*, $\mathbb{E}_{\boldsymbol{\theta}}$ the expectation under $P_{\boldsymbol{\theta}}(.|x)$, and $z_C(\mathbf{y}_C)$ the component of $\mathbf{z} \in \mathcal{Z}(x)$ indexed by the configuration $\mathbf{y}_C$ of factor C.

Proposition 1 *There is a map coupling each distribution $P_{\boldsymbol{\theta}}(.|x) \in \mathcal{P}$ to a unique $\mathbf{z} \in \mathcal{Z}(x)$ such that $\mathbb{E}_{\boldsymbol{\theta}}[\chi(Y)] = \mathbf{z}$. Define $H(\mathbf{z}) \triangleq H(P_{\boldsymbol{\theta}}(.|x))$ if some $P_{\boldsymbol{\theta}}(.|x)$ is coupled to $\mathbf{z}$, and $H(\mathbf{z}) = -\infty$ if no such $P_{\boldsymbol{\theta}}(.|x)$ exists. Then:*

1. *The following variational representation for the log-partition function holds:*

$$\log Z(\boldsymbol{\theta}, x) = \max_{\mathbf{z} \in \mathcal{Z}(x)} \boldsymbol{\theta}^\top \mathbf{F}(x)\mathbf{z} + H(\mathbf{z}). \tag{6}$$

2. *The problem in (6) is convex and its solution is attained at the factor marginals, i.e., there is a maximizer $\bar{\mathbf{z}}$ s.t. $\bar{z}_C(\mathbf{y}_C) = \Pr_{\boldsymbol{\theta}}\{Y_C = \mathbf{y}_C\}$ for each $C \in \mathcal{C}$. The gradient of the log-partition function is $\nabla \log Z(\boldsymbol{\theta}, x) = \mathbf{F}(x)\bar{\mathbf{z}}$.*
3. *The MAP $\hat{y} \triangleq \text{argmax}_{y \in \mathcal{Y}(x)} P_{\boldsymbol{\theta}}(y|x)$ can be obtained by solving the linear program*

$$\hat{\mathbf{z}} \triangleq \chi(\hat{y}) = \operatorname*{argmax}_{\mathbf{z} \in \mathcal{Z}(x)} \boldsymbol{\theta}^\top \mathbf{F}(x)\mathbf{z}. \tag{7}$$

Proof: [Wainwright and Jordan, 2008, Theorem 3.4] provide a proof for the canonical overcomplete representation where $\mathbf{F}(x)$ is the identity matrix, *i.e.*, each feature is an indicator of the configuration of the factor. In that case, the map from the parameter space to the relative interior of the marginal polytope is surjective. In our model, arbitrary features are allowed and the parameters are *tied*, since they are shared by all factors. This can be expressed as a linear map

[2] The marginal polytope can also be defined as the set of factor marginals realizable by distributions that factor according to the graph. Log-linear models with *canonical overcomplete parametrization*—*i.e.*, whose sufficient statistics (features) at each factor are configuration indicators—are studied in Wainwright and Jordan [2008].

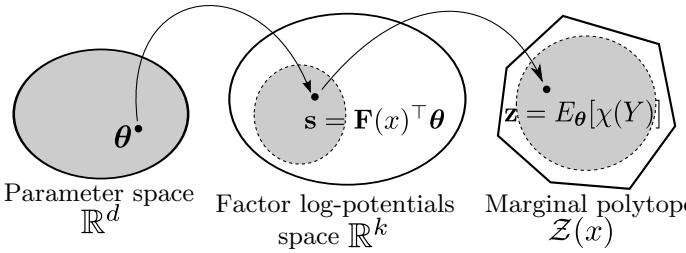

Figure 2: Dual parametrization of the distributions in $\mathcal{P}$. The original parameter is linearly mapped to the factor log-potentials, the *canonical overcomplete parameter space* Wainwright and Jordan [2008], which is mapped onto the relative interior of the marginal polytope $\mathcal{Z}(x)$. In general only a subset of $\mathcal{Z}(x)$ is reachable from our parameter space.

$\boldsymbol{\theta} \mapsto \mathbf{s} = \mathbf{F}(x)^\top \boldsymbol{\theta}$ that "places" our parameters $\boldsymbol{\theta} \in \mathbb{R}^d$ onto a *linear subspace* of the canonical overcomplete parameter space; therefore, our map $\boldsymbol{\theta} \mapsto \mathbf{z}$ is not necessarily onto $\mathrm{ri}\mathcal{Z}(x)$, unlike in Wainwright and Jordan [2008], and our $H(\mathbf{z})$ is defined slightly differently: it can take the value $-\infty$ if no $\boldsymbol{\theta}$ maps to $\mathbf{z}$. This does not affect the expression in (6), since the solution of this optimization problem with our $H(\mathbf{z})$ replaced by theirs is also the feature expectation under $P_{\boldsymbol{\theta}}(.|x)$ and the associated $\mathbf{z}$, by definition, always yields a finite $H(\mathbf{z})$. ■

3.2 Loss Evaluation and Differentiation

We now invoke Prop. 1 to derive a variational expression for evaluating any loss $L_{\beta,\gamma}(\boldsymbol{\theta}; x, y)$ in (3), and compute its gradient as a by-product.[3] This is crucial for the learning algorithms to be introduced in §4. Our only assumption is that the cost function $\ell(y', y)$ can be written as a sum over factor-local costs; letting $\mathbf{z} = \chi(y)$ and $\mathbf{z}' = \chi(y')$, this implies $\ell(y', y) = \mathbf{p}^\top \mathbf{z}' + q$ for some $\mathbf{p}$ and q which are constant with respect to $\mathbf{z}'$.[4] Under this assumption, and letting $\mathbf{s} = \mathbf{F}(x)^\top \boldsymbol{\theta}$ be the vector of factor log-potentials, $L_{\beta,\gamma}(\boldsymbol{\theta}; x, y)$ becomes expressible in terms of the log-partition function of a distribution whose log-potentials are set to $\beta(\mathbf{s} + \gamma\mathbf{p})$. From (6), we obtain

$$L_{\beta,\gamma}(\boldsymbol{\theta}; x, y) = \max_{\mathbf{z}' \in \mathcal{Z}(x)} \boldsymbol{\theta}^\top \mathbf{F}(x)(\mathbf{z}' - \mathbf{z}) + \frac{1}{\beta} H(\mathbf{z}') + \gamma(\mathbf{p}^\top \mathbf{z}' + q). \tag{8}$$

Let $\bar{\mathbf{z}}$ be a maximizer in (8); from the second statement of Prop. 1 we obtain the following expression for the gradient of $L_{\beta,\gamma}$ at $\boldsymbol{\theta}$:

$$\nabla L_{\beta,\gamma}(\boldsymbol{\theta}; x, y) = \mathbf{F}(x)(\bar{\mathbf{z}} - \mathbf{z}). \tag{9}$$

For concreteness, we revisit the examples discussed in the previous subsection.

[3] Our description also applies to the (non-differentiable) hinge loss case, when $\beta \to \infty$, if we replace all instances of "the gradient" in the text by "a subgradient."

[4] For the Hamming loss, this holds with $\mathbf{p} = 1 - 2\mathbf{z}$ and $q = \mathbf{1}^\top \mathbf{z}$. See Taskar et al. [2006] for other examples.

Sequence Labeling. Without hard constraints, the graphical model does not contain loops, and therefore $L_{\beta,\gamma}(\boldsymbol{\theta}; x, y)$ and $\nabla L_{\beta,\gamma}(\boldsymbol{\theta}; x, y)$ may be easily computed by setting the log-potentials as described above and running the forward-backward algorithm.

Dependency Parsing. For the arc-factored model, $L_{\beta,\gamma}(\boldsymbol{\theta}; x, y)$ and $\nabla L_{\beta,\gamma}(\boldsymbol{\theta}; x, y)$ may be computed exactly by modifying the log-potentials, invoking the matrix-tree theorem to compute the log-partition function and the marginals [Smith and Smith, 2007, Koo et al., 2007, McDonald and Satta, 2007], and using the fact that $H(\bar{\mathbf{z}}) = \log Z(\boldsymbol{\theta}, x) - \boldsymbol{\theta}^\top \mathbf{F}(x)\bar{\mathbf{z}}$. The marginal polytope is the same as the *arborescence polytope* in Martins et al. [2009]. For richer models where arc interactions are considered, exact inference is intractable. Both the marginal polytope and the entropy, necessary in (6), lack concise closed form expressions. Two approximate approaches have been recently proposed: a loopy belief propagation (BP) algorithm for computing pseudo-marginals [Smith and Eisner, 2008]; and an LP-relaxation method for approximating the most likely parse tree [Martins et al., 2009]. Although the two methods may look unrelated at first sight, both optimize over outer bounds of the marginal polytope. See [Martins et al., 2010] for further discussion.

4 Online Learning

We now propose a dual coordinate ascent approach to learn the model parameters $\boldsymbol{\theta}$. This approach extends the primal-dual view of online algorithms put forth by Shalev-Shwartz and Singer [2006] to structured classification; it handles any loss in (3). In the case of the hinge loss, we recover the online passive-aggressive algorithm (also known as MIRA, [Crammer et al., 2006]) as well as its k-best variants. With the logistic loss, we obtain a new passive-aggressive algorithm for CRFs.

Start by noting that the learning problem in (2) is not affected if we multiply the objective by m. Consider a sequence of primal objectives $P_1(\boldsymbol{\theta}), \ldots, P_{m+1}(\boldsymbol{\theta})$ to be minimized, each of the form

$$P_t(\boldsymbol{\theta}) = \lambda m R(\boldsymbol{\theta}) + \sum_{i=1}^{t-1} L(\boldsymbol{\theta}; x_i, y_i).$$

Our goal is to minimize $P_{m+1}(\boldsymbol{\theta})$; for simplicity we consider online algorithms with only one pass over the data, but the analysis can be extended to the case where multiple epochs are allowed.

Below, we let $\bar{\mathbb{R}} \triangleq \mathbb{R} \cup \{+\infty\}$ be the extended reals and, given a function $f : \mathbb{R}^n \to \bar{\mathbb{R}}$, we denote by $f^\star : \mathbb{R}^n \to \bar{\mathbb{R}}$ its *convex conjugate*, $f^\star(\mathbf{y}) = \sup_{\mathbf{x}} \mathbf{x}^\top \mathbf{y} - f(\mathbf{x})$ (see Appendix A for a background of convex analysis). The next proposition, proved in [Kakade and Shalev-Shwartz, 2008], states a generalized form of Fenchel duality, which involves a dual vector $\boldsymbol{\mu}_i \in \mathbb{R}^d$ per each instance.

Proposition 2 ([Kakade and Shalev-Shwartz, 2008]) *The Lagrange dual of* $\min_{\boldsymbol{\theta}} P_t(\boldsymbol{\theta})$ *is*

$$\max_{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{t-1}} D_t(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{t-1}),$$

where

$$D_t(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{t-1}) = -\lambda m R^\star \left(-\frac{1}{\lambda m} \sum_{i=1}^{t-1} \boldsymbol{\mu}_i \right) - \sum_{i=1}^{t-1} L^\star(\boldsymbol{\mu}_i; x_i, y_i). \tag{10}$$

Algorithm 1 Dual coordinate ascent (DCA)

Input: $\mathcal{D}$, λ, number of iterations K
Initialize $\boldsymbol{\theta}_1 = \mathbf{0}$; set $m = |\mathcal{D}|$ and $T = mK$
for $t = 1$ **to** T **do**
  Receive an instance x_t, y_t
  Update $\boldsymbol{\theta}_{t+1}$ by solving (11) exactly or approximately (see Alg. 2)
end for
Return the averaged model $\bar{\boldsymbol{\theta}} \leftarrow \frac{1}{T}\sum_{t=1}^{T}\boldsymbol{\theta}_t$.

Algorithm 2 Parameter updates

Input: current model $\boldsymbol{\theta}_t$, instance (x_t, y_t), λ

Obtain $\mathbf{z}_t$ from y_t
Solve the variational problem in (8) to obtain $\bar{\mathbf{z}}_t$ and $L_{\beta,\gamma}(\boldsymbol{\theta}_t, x_t, y_t)$
Compute $\nabla L_{\beta,\gamma}(\boldsymbol{\theta}_t, x_t, y_t) = \mathbf{F}(x_t)(\bar{\mathbf{z}}_t - \mathbf{z}_t)$
Compute $\eta_t = \min\left\{\frac{1}{\lambda m}, \frac{L(\boldsymbol{\theta}_t; x_t, y_t)}{\|\nabla L(\boldsymbol{\theta}_t; x_t, y_t)\|^2}\right\}$
Return $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \nabla L(\boldsymbol{\theta}_t; x_t, y_t)$

If $R(\boldsymbol{\theta}) = \frac{1}{2}\|\boldsymbol{\theta}\|^2$, then $R = R^\star$, and strong duality holds for any convex L, i.e., $P_t(\boldsymbol{\theta}^*) = D_t(\boldsymbol{\mu}_1^*, \ldots, \boldsymbol{\mu}_{t-1}^*)$ where $\boldsymbol{\theta}^*$ and $\boldsymbol{\mu}_1^*, \ldots, \boldsymbol{\mu}_{t-1}^*$ are respectively the primal and dual optima. Moreover, the following primal-dual relation holds: $\boldsymbol{\theta}^* = -\frac{1}{\lambda m}\sum_{i=1}^{t-1}\boldsymbol{\mu}_i^*$.

We can therefore transform our problem into that of maximizing $D_{m+1}(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_m)$. Dual coordinate ascent (DCA) is an umbrella name for algorithms that manipulate a single dual coordinate at a time. In our setting, the largest such improvement at round t is achieved by $\boldsymbol{\mu}_t \triangleq \operatorname{argmax}_{\boldsymbol{\mu}} D_{t+1}(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{t-1}, \boldsymbol{\mu})$. The next proposition, proved in Appendix B, characterizes the mapping of this subproblem back into the primal space, shedding light on the connections with known online algorithms.

Proposition 3 *Let $\boldsymbol{\theta}_t \triangleq -\frac{1}{\lambda m}\sum_{i=1}^{t-1}\boldsymbol{\mu}_i$. The Lagrange dual of $\max_{\boldsymbol{\mu}} D_{t+1}(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{t-1}, \boldsymbol{\mu})$ is*

$$\min_{\boldsymbol{\theta}} \frac{\lambda m}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}_t\|^2 + L(\boldsymbol{\theta}; x_t, y_t). \tag{11}$$

Assembling these pieces together yields Alg. 1, where the solution of (11) is carried out by Alg. 2, as explained next.[5] While the problem in (11) is easier than the batch problem in (2), an exact solution may still be prohibitively expensive in large-scale settings, particularly because it has to be solved repeatedly. We thus adopt a simpler strategy that still guarantees some improvement in the dual. Noting that L is non-negative, we may rewrite (11) as

$$\min_{\boldsymbol{\theta},\xi} \tfrac{\lambda m}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}_t\|^2 + \xi \quad \text{s.t.} \quad L(\boldsymbol{\theta}; x_t, y_t) \le \xi, \quad \xi \ge 0. \tag{12}$$

From the convexity of L, we may take its first-order Taylor approximation around $\boldsymbol{\theta}_t$ to obtain the lower bound $L(\boldsymbol{\theta}; x_t, y_t) \ge L(\boldsymbol{\theta}_t; x_t, y_t) + (\boldsymbol{\theta} - \boldsymbol{\theta}_t)^\top \nabla L(\boldsymbol{\theta}_t; x_t, y_t)$. Therefore the true minimum in (11) is lower bounded by

$$\begin{aligned} \min_{\boldsymbol{\theta},\xi} \quad & \tfrac{\lambda m}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}_t\|^2 + \xi \\ \text{s.t.} \quad & L(\boldsymbol{\theta}_t; x_t, y_t) + (\boldsymbol{\theta} - \boldsymbol{\theta}_t)^\top \nabla L(\boldsymbol{\theta}_t; x_t, y_t) \le \xi, \quad \xi \ge 0. \end{aligned} \tag{13}$$

[5] The final averaging step in Alg. 1 is an online-to-batch conversion with good generalization guarantees Cesa-Bianchi et al. [2004].

This is a Euclidean projection problem (with slack) that admits the closed form solution $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \nabla L(\boldsymbol{\theta}_t; x_t, y_t)$, with

$$\eta_t = \min \left\{ \tfrac{1}{\lambda m}, \tfrac{L(\boldsymbol{\theta}_t; x_t, y_t)}{\|\nabla L(\boldsymbol{\theta}_t; x_t, y_t)\|^2} \right\}. \tag{14}$$

Example: 1-best MIRA. If L is the hinge-loss, we obtain from (9) $\nabla L_{\text{SVM}}(\boldsymbol{\theta}_t; x_t, y_t) = \mathbf{F}(x_t)(\bar{\mathbf{z}}_t - \mathbf{z}_t) = \boldsymbol{\phi}(x_t, \hat{y}_t) - \boldsymbol{\phi}(x_t, y_t)$, where $\hat{y}_t = \operatorname{argmax}_{y'_t \in \mathcal{Y}(x_t)} \boldsymbol{\theta}_t^\top (\boldsymbol{\phi}(x_t, y'_t) - \boldsymbol{\phi}(x_t, y_t)) + \ell(y'_t, y_t)$. The update becomes $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t(\boldsymbol{\phi}(x_t, \hat{y}_t) - \boldsymbol{\phi}(x_t, y_t))$, with

$$\eta_t \;=\; \min \left\{ \frac{1}{\lambda m}, \frac{\boldsymbol{\theta}_t^\top (\boldsymbol{\phi}(x_t, \hat{y}_t) - \boldsymbol{\phi}(x_t, y_t)) + \ell(\hat{y}_t, y_t)}{\|\boldsymbol{\phi}(x_t, \hat{y}_t) - \boldsymbol{\phi}(x_t, y_t)\|^2} \right\}. \tag{15}$$

This is precisely the max-loss variant of the 1-best MIRA algorithm [Crammer et al., 2006, §8]. Hence, while MIRA was originally motivated by a conservativeness-correctness tradeoff, it turns out that it also performs coordinate ascent in the dual.

Example: CRFs. This framework immediately allows us to extend 1-best MIRA for CRFs, which optimizes the logistic loss. In that case, the exact problem in (12) can be expressed as

$$\min_{\boldsymbol{\theta}, \xi} \tfrac{\lambda m}{2} \|\boldsymbol{\theta} - \boldsymbol{\theta}_t\|^2 + \xi \quad \text{s.t.} \quad -\log P_{\boldsymbol{\theta}}(y_t | x_t) \leq \xi, \quad \xi \geq 0.$$

In words: stay as close as possible to the previous parameter vector, but correct the model so that the conditional probability $P_{\boldsymbol{\theta}}(y_t|x_t)$ becomes large enough. From (9), $\nabla L_{\text{CRF}}(\boldsymbol{\theta}_t; x_t, y_t) = \mathbf{F}(x_t)(\bar{\mathbf{z}}_t - \mathbf{z}_t) = \mathbb{E}_{\boldsymbol{\theta}_t} \boldsymbol{\phi}(x_t, Y_t) - \boldsymbol{\phi}(x_t, y_t)$, where now $\bar{\mathbf{z}}_t$ is an expectation instead of a mode. The update becomes $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t(\mathbb{E}_{\boldsymbol{\theta}_t} \boldsymbol{\phi}(x_t, Y_t) - \boldsymbol{\phi}(x_t, y_t))$, with

$$\eta_t = \min \left\{ \tfrac{1}{\lambda m}, \tfrac{\boldsymbol{\theta}_t^\top (\mathbb{E}_{\boldsymbol{\theta}_t} \boldsymbol{\phi}(x_t, Y_t) - \boldsymbol{\phi}(x_t, y_t)) + H(P_{\boldsymbol{\theta}_t}(.|x_t))}{\|\mathbb{E}_{\boldsymbol{\theta}_t} \boldsymbol{\phi}(x_t, Y_t) - \boldsymbol{\phi}(x_t, y_t)\|^2} \right\} = \min \left\{ \tfrac{1}{\lambda m}, \tfrac{-\log P_{\boldsymbol{\theta}_t}(y_t|x_t)}{\|\mathbb{E}_{\boldsymbol{\theta}_t} \boldsymbol{\phi}(x_t, Y_t) - \boldsymbol{\phi}(x_t, y_t)\|^2} \right\}. \tag{16}$$

Thus, the difference with respect to standard 1-best MIRA (15) consists of replacing the feature vector of the loss-augmented mode $\boldsymbol{\phi}(x_t, \hat{y}_t)$ by the expected feature vector $\mathbb{E}_{\boldsymbol{\theta}_t} \boldsymbol{\phi}(x_t, Y_t)$ and the cost function $\ell(\hat{y}_t, y_t)$ by the entropy function $H(P_{\boldsymbol{\theta}_t}(.|x_t))$.

Example: k-best MIRA. Tighter approximations to the problem in (11) can be built by using the variational representation machinery; see (8) for losses in the family $L_{\beta,\gamma}$. Plugging this variational representation into the constraint in (12) we obtain the following semi-infinite quadratic program:

$$\begin{array}{ll} \min\limits_{\boldsymbol{\theta}, \xi} & \tfrac{\lambda m}{2} \|\boldsymbol{\theta} - \boldsymbol{\theta}_t\|^2 + \xi \\ \text{s.t.} & \boldsymbol{\theta} \in \mathcal{H}(\mathbf{z}'_t; \beta, \gamma), \quad \forall \mathbf{z}'_t \in \mathcal{Z}(x) \\ & \xi \geq 0. \end{array} \tag{17}$$

where $\mathcal{H}(\mathbf{z}'; \mathbf{z}, \beta, \gamma) \triangleq \{\boldsymbol{\theta} \mid \mathbf{a}^\top \boldsymbol{\theta} \leq b\}$ is a half-space with $\mathbf{a} = \mathbf{F}(x)(\mathbf{z}' - \mathbf{z})$ and $b = \xi - \gamma(\mathbf{p}^\top \mathbf{z}' + q) - \beta^{-1} H(\mathbf{z}')$. The constraint set in (17) is a convex set defined by the intersection of uncountably

many half-spaces (indexed by the points in the marginal polytope).[6] Our approximation consisted of relaxing the problem in (17) by discarding all half-spaces except the one indexed by $\bar{\mathbf{z}}_t$, the dual parameter of the current iterate $\boldsymbol{\theta}_t$; however, tigher relaxations are obtained by keeping some of the other half-spaces. For the hinge loss, rather than just using the mode $\bar{\mathbf{z}}_t$, one may rank the k-best outputs and add a half-space constraint for each. This procedure approximates the constraint set by a polyhedron and the resulting problem can be addressed using row-action methods, such as Hildreth's algorithm [Censor and Zenios, 1997]. This corresponds precisely to k-best MIRA.[7]

5 Experiments

We report experiments on two tasks: named entity recognition and dependency parsing. For each, we compare DCA (Alg. 1) with SGD. We report results for several values for the regularization parameter $C = 1/(\lambda m)$. To choose the learning rate for SGD, we use the formula $\eta_t = \eta/(1 + (t-1)/m)$ [LeCun et al., 1998]. We choose η using dev-set validation after a single epoch [Collins et al., 2008].

Named Entity Recognition. We use the English data from the CoNLL 2003 shared task [Tjong Kim Sang and De Meulder, 2003], which consist of English news articles annotated with four entity types: person, location, organization, and miscellaneous. We used a standard set of feature templates, as in [Kazama and Torisawa, 2007], with token shape features [Collins, 2002b] and simple gazetteer features; a feature was included iff it occurs at least once in the training set (total 1,312,255 features). The task is evaluated using the F_1 measure computed at the granularity of entire entities. We set $\beta = 1$ and $\gamma = 0$ (the CRF case). In addition to SGD, we also compare with L-BFGS [Liu and Nocedal, 1989], a common choice for optimizing conditional log-likelihood. We used $\{10^a, a = -3, \ldots, 2\}$ for the set of values considered for η in SGD. Fig. 3 shows that DCA (which only requires tuning one hyperparameter) reaches better-performing models than the baselines.

Dependency Parsing. We trained non-projective dependency parsers for three languages (Arabic, Danish, and English), using datasets from the CoNLL-X and CoNLL-2008 shared tasks [Buchholz and Marsi, 2006, Surdeanu et al., 2008]. Performance is assessed by the unlabeled attachment score (UAS), the fraction of non-punctuation words which were assigned the correct parent. We adapted TurboParser[8] to handle any loss function $L_{\beta,\gamma}$ via Alg. 1; for decoding, we used the loopy BP algorithm of Smith and Eisner [2008] (see §3.2). We used the pruning strategy in [Martins et al., 2009] and tried two feature configurations: an arc-factored model, for which decoding is exact, and a model with second-order features (siblings and grandparents) for which it is approximate. The comparison with SGD for the CRF case is shown in Fig. 4. For the arc-factored models,

[6] Interestingly, when the hinge loss is used, only a finite (albeit exponentially many) of these half-spaces are necessary, those indexed by *vertices* of the marginal polytope. In this case, the constraint set is polyhedral.

[7] The prediction-based variant of 1-best MIRA [Crammer et al., 2006] is also a particular case, where $\bar{\mathbf{z}}_t$ is the prediction under the current model $\boldsymbol{\theta}_t$, rather than the mode of $L_{\text{SVM}}(\boldsymbol{\theta}_t, x_t, y_t)$.

[8] Available at `http://www.ark.cs.cmu.edu/TurboParser`.

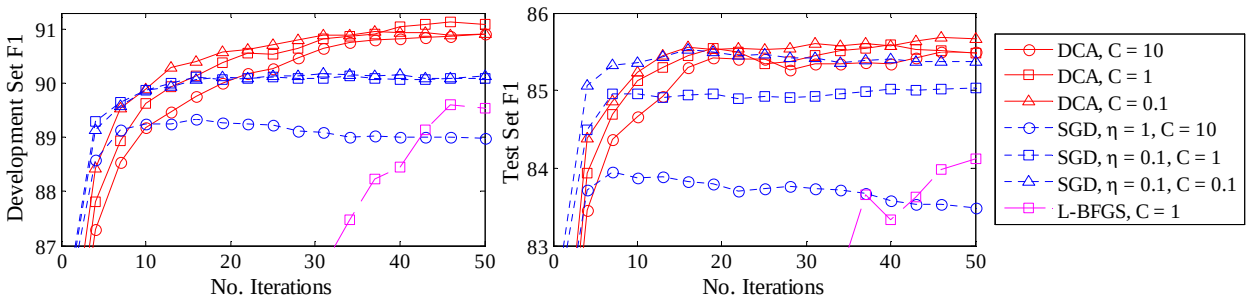

Figure 3: Named entity recognition. Learning curves for DCA (Alg. 1), SGD, and L-BFGS. The SGD curve for $C = 10$ is lower than the others because dev-set validation chose a suboptimal value of η. DCA, by contrast, does not require choosing any hyperparameters other than C. L-BFGS ultimately converges after 121 iterations to an F_1 of 90.53 on the development data and 85.31 on the test data.

	β	1	1	1	1	3	5	∞
	γ	0 (CRF)	1	3	5	1	1	1 (SVM)
NER	BEST C	1.0	10.0	1.0	1.0	1.0	1.0	1.0
	F_1 (%)	85.48	85.54	85.65	85.72	85.55	85.48	85.41
DEPENDENCY PARSING	BEST C	0.1	0.01	0.01	0.01	0.01	0.01	0.1
	UAS (%)	90.76	90.95	91.04	91.01	90.94	90.91	90.75

Table 1: Varying β and γ: neither the CRF nor the SVM are optimal. We report only the results for the best C, chosen from $\{0.001, 0.01, 0.1, 1\}$ with dev-set validation. For named entity recognition, we show test set F_1 after $K = 50$ iterations (empty cells will be filled in in the final version). Dependency parsing experiments used the arc-factored model on English and $K = 10$.

the learning curve of DCA seems to lead faster to an accurate model. Notice that the plots do not account for the fact that SGD requires four extra iterations to choose the learning rate. For the second-order models of Danish and English, however, DCA did not perform as well.[9]

Finally, Table 1 shows results obtained for different settings of β and γ.[10] Interestingly, we observe that the higher scores are obtained for loss functions that are "between" SVMs and CRFs.

6 Conclusion

We presented a general framework for aggressive online learning of structured classifiers by optimizing any loss function in a wide family. The technique does not require a learning rate to be specified. We derived an efficient technique for evaluating the loss function and its gradient. Exper-

[9]Further analysis showed that for ∼15% of the training instances, loopy BP led to very poor variational approximations of $\log Z(\boldsymbol{\theta}, x)$, yielding estimates $P_{\boldsymbol{\theta}_t}(y_t|x_t) > 1$, thus a negative learning rate (see (16)), that we truncate to zero. Thus, no update occurs for those instances, explaining the slower convergence. A possible way to fix this problem is to use techniques that guarantee upper bounds on the log-partition function Wainwright and Jordan [2008].

[10]Observe that there are only two degrees of freedom: indeed, (λ, β, γ) and $(\lambda', \beta', \gamma')$ lead to equivalent learning problems if $\lambda' = \lambda/a$, $\beta' = \beta/a$ and $\gamma' = a\gamma$ for any $a > 0$, with the solutions related via $\boldsymbol{\theta}' = a\boldsymbol{\theta}$.

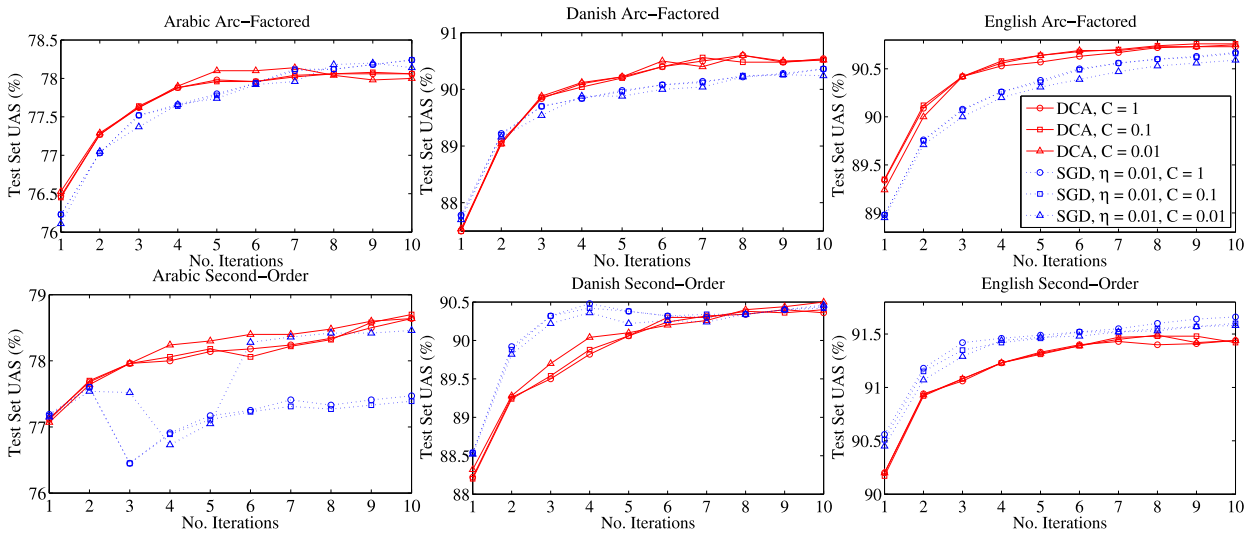

Figure 4: Learning curves for DCA (Alg. 1) and SGD, the latter with the learning rate $\eta = 0.01$ chosen from $\{0.001, 0.01, 0.1, 1\}$ using the same procedure as before. The instability when training the second-order models might be due to the fact that inference there is approximate.

iments in named entity recognition and dependency parsing showed that the algorithm converges to accurate models at least as fast as stochastic gradient descent.

References

Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden Markov support vector machines. In *ICML*, 2003.

S. P. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

S. Buchholz and E. Marsi. CoNLL-X shared task on multilingual dependency parsing. In *CoNLL*, 2006.

Y. Censor and S. A. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, 1997.

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

M. Collins. Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In *EMNLP*, 2002a.

M. Collins. Ranking algorithms for named-entity extraction: Boosting and the voted perceptron. In *ACL*, 2002b.

M. Collins, A. Globerson, T. Koo, X. Carreras, and P.L. Bartlett. Exponentiated gradient algorithms for conditional random fields and max-margin Markov networks. *JMLR*, 2008.

R. Collobert, F. Sinz, J. Weston, and L. Bottou. Trading convexity for scalability. In *ICML*, 2006.

K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *JMLR*, 7:551–585, 2006.

J. Eisner. Three new probabilistic models for dependency parsing: An exploration. In *COLING*, 1996.

K. Gimpel and N. A. Smith. Softmax-margin CRFs: Training log-linear models with cost functions. In *NAACL*, 2010.

S. Kakade and S. Shalev-Shwartz. Mind the duality gap: Logarithmic regret algorithms for online optimization. In *NIPS*, 2008.

J. Kazama and K. Torisawa. A new perceptron algorithm for sequence labeling with non-local features. In *Proc. of EMNLP-CoNLL*, 2007.

T. Koo, A. Globerson, X. Carreras, and M. Collins. Structured prediction models via the matrix-tree theorem. In *EMNLP*, 2007.

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.

S. Kübler, R. McDonald, and J. Nivre. *Dependency Parsing*. Morgan & Claypool, 2009.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML*, 2001.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Math. Programming*, 45:503–528, 1989.

A. F. T. Martins, N. A. Smith, and E. P. Xing. Concise integer linear programming formulations for dependency parsing. In *Proc. of ACL*, 2009.

A. F. T. Martins, N. A. Smith, E. P. Xing, P. M. Q. Aguiar, and M. A. T. Figueiredo. Turbo parsers: Dependency parsing by approximate variational inference. In *Proc. of EMNLP*, 2010.

A. McCallum, K. Schultz, and S. Singh. Factorie: Probabilistic programming via imperatively defined factor graphs. In *NIPS*, 2009.

R. McDonald and G. Satta. On the complexity of non-projective data-driven dependency parsing. In *IWPT*, 2007.

R. T. McDonald, F. Pereira, K. Ribarov, and J. Hajic. Non-projective dependency parsing using spanning tree algorithms. In *Proc. of HLT-EMNLP*, 2005.

S. Shalev-Shwartz and Y. Singer. Online learning meets optimization in the dual. In *COLT*, 2006.

D. A. Smith and J. Eisner. Minimum risk annealing for training log-linear models. In *ACL*, 2006.

D. A. Smith and J. Eisner. Dependency parsing by belief propagation. In *EMNLP*, 2008.

D. A. Smith and N. A. Smith. Probabilistic models of nonprojective dependency trees. In *EMNLP*, 2007.

M. Surdeanu, R. Johansson, A. Meyers, L. Màrquez, and J. Nivre. The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. *Proc. of CoNLL*, 2008.

B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *NIPS*, 2003.

B. Taskar, S. Lacoste-Julien, and M. I. Jordan. Structured prediction, dual extragradient and Bregman projections. *JMLR*, 7:1627–1653, 2006.

E. F. Tjong Kim Sang and F. De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proc. of CoNLL*, 2003.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *Proc. of ICML*, 2004.

M. J. Wainwright and M. I. Jordan. *Graphical Models, Exponential Families, and Variational Inference*. Now Publishers, 2008.

A Background on Convex Analysis

We briefly review some notions of convex analysis that are used throughout the paper. For more details, see *e.g.* Boyd and Vandenberghe [2004]. Below, $\Delta^d \triangleq \{\boldsymbol{\mu} \in \mathbb{R}^d \mid \sum_{j=1}^d \mu_j = 1, \mu_j \geq 0 \ \forall j\}$ is the probability simplex in $\mathbb{R}^d$, and $\mathcal{B}_\gamma(\mathbf{x}) = \{\mathbf{y} \in \mathbb{R}^d \mid \|\mathbf{y} - \mathbf{x}\| \leq \gamma\}$ is the ball with radius γ centered at $\mathbf{x}$.

A set $\mathcal{C} \subseteq \mathbb{R}^d$ is *convex* if $\mu\mathbf{x} + (1-\mu)\mathbf{y} \in \mathcal{C}$ for all $\mathbf{x}, \mathbf{y} \in \mathcal{C}$ and $\mu \in [0,1]$. The *convex hull* of a set $\mathcal{X} \subseteq \mathbb{R}^d$ is the set of all convex combinations of the elements of $\mathcal{X}$,

$$\operatorname{conv} \mathcal{X} = \left\{ \sum_{i=1}^p \mu_i \mathbf{x}_i \;\middle|\; \boldsymbol{\mu} \in \Delta^p, p \geq 1 \right\};$$

it is also the smallest convex set that contains $\mathcal{X}$. The *affine hull* of $\mathcal{X} \subseteq \mathbb{R}^d$ is the set of all affine combinations of the elements of $\mathcal{X}$,

$$\operatorname{aff} \mathcal{X} = \left\{ \sum_{i=1}^p \mu_i \mathbf{x}_i \;\middle|\; \sum_{j=1}^p \mu_j = 1, p \geq 1 \right\};$$

it is also the smallest affine set that contains $\mathcal{X}$. The *relative interior* of $\mathcal{X}$ is its interior relative to the affine hull $\mathcal{X}$,

$$\operatorname{relint} \mathcal{X} = \{\mathbf{x} \in \mathcal{X} \mid \exists \gamma > 0 \ : \ \mathcal{B}_\gamma(\mathbf{x}) \cap \operatorname{aff} \mathcal{X} \subseteq \mathcal{X}\}.$$

Let $\bar{\mathbb{R}} \triangleq \mathbb{R} \cup \{+\infty\}$ be the extended reals. The *effective domain* of a function $f : \mathbb{R}^d \to \bar{\mathbb{R}}$ is the set $\operatorname{dom} f = \{\mathbf{x} \in \mathbb{R}^d \mid f(\mathbf{x}) < +\infty\}$. f is *proper* if $\operatorname{dom} f \neq \varnothing$. The epigraph of f is the set $\operatorname{epi} f \triangleq \{(x,t) \in \mathbb{R}^d \times \mathbb{R} \,|\, f(x) \leq t\}$. f is *lower semicontinuous* (lsc) if the epigraph is closed in $\mathbb{R}^d \times \mathbb{R}$. f is *convex* if $\operatorname{dom} f$ is a convex set and

$$f(\mu\mathbf{x} + (1-\mu)\mathbf{y}) \leq \mu f(\mathbf{x}) + (1-\mu) f(\mathbf{y}), \quad \forall \mathbf{x}, \mathbf{y} \in \operatorname{dom} f, \quad \mu \in [0,1].$$

The (Fenchel) *conjugate* of f is the function $f^\star : \mathbb{R}^d \to \bar{\mathbb{R}}$ defined as

$$f^\star(\mathbf{y}) = \sup_{\mathbf{x} \in \mathbb{R}^d} \mathbf{x}^\top \mathbf{y} - f(\mathbf{x}).$$

$f^\star$ is always convex, since it is the supremum of a family of affine functions. Some examples follow:

- If f is an affine function, $f(\mathbf{x}) = \mathbf{a}^\top \mathbf{x} + b$, then $f^\star(\mathbf{y}) = -b$ if $\mathbf{y} = \mathbf{a}$ and $-\infty$ otherwise.
- If f is the ℓ_p-norm, $f(\mathbf{x}) = \|\mathbf{x}\|_p$, then $f^\star$ is the indicator of the unit ball induced by the dual norm, $f^\star(\mathbf{y}) = 0$ if $\|\mathbf{y}\|_q \leq 1$ and $+\infty$ otherwise, with $p^{-1} + q^{-1} = 1$.
- If f is half of the squared ℓ_p-norm, $f(\mathbf{x}) = \|\mathbf{x}\|_p^2/2$, then $f^\star$ is half of the squared dual norm, $f^\star(\mathbf{y}) = \|\mathbf{y}\|_q^2/2$, with $p^{-1} + q^{-1} = 1$.
- If f is convex, lsc, and proper, then $f^{\star\star} = f$.
- If $g(\mathbf{x}) = tf(\mathbf{x} - \mathbf{x}_0)$, with $t \in \mathbb{R}_+$ and $\mathbf{x}_0 \in \mathbb{R}^d$, then $g^\star(\mathbf{y}) = \mathbf{x}_0^\top \mathbf{y} + tf^\star(\mathbf{y}/t)$.

B Proof of Proposition 3

From (10),

$$
\begin{aligned}
&\max_{\boldsymbol{\mu}} D_{t+1}(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{t-1}, \boldsymbol{\mu}) \\
&= \max_{\boldsymbol{\mu}} -\frac{1}{2\lambda m} \left\| \sum_{i=1}^{t-1} \boldsymbol{\mu}_i + \boldsymbol{\mu} \right\|^2 - L^\star(\boldsymbol{\mu}; x_t, y_t) - \sum_{i=1}^{t-1} L^\star(\boldsymbol{\mu}_i; x_i, y_i) \\
&= \max_{\boldsymbol{\mu}} -\frac{1}{2\lambda m} \left\| -\lambda m \boldsymbol{\theta}_t + \boldsymbol{\mu} \right\|^2 - L^\star(\boldsymbol{\mu}; x_t, y_t) + \text{constant} \\
&=^{\text{(i)}} \max_{\boldsymbol{\mu}} -\frac{1}{2\lambda m} \left\| -\lambda m \boldsymbol{\theta}_t + \boldsymbol{\mu} \right\|^2 - \max_{\boldsymbol{\theta}} (\boldsymbol{\mu}^\top \boldsymbol{\theta} - L(\boldsymbol{\theta}; x_t, y_t)) + \text{constant} \\
&=^{\text{(ii)}} \min_{\boldsymbol{\theta}} \max_{\boldsymbol{\mu}} -\frac{1}{2\lambda m} \left\| -\lambda m \boldsymbol{\theta}_t + \boldsymbol{\mu} \right\|^2 - \boldsymbol{\mu}^\top \boldsymbol{\theta} + L(\boldsymbol{\theta}; x_t, y_t) + \text{constant} \\
&= \min_{\boldsymbol{\theta}} \left(\max_{\boldsymbol{\mu}} \boldsymbol{\mu}^\top (-\boldsymbol{\theta}) - \frac{1}{2\lambda m} \left\| \boldsymbol{\mu} - \lambda m \boldsymbol{\theta}_t \right\|^2 \right) + L(\boldsymbol{\theta}; x_t, y_t) + \text{constant} \\
&=^{\text{(iii)}} \min_{\boldsymbol{\theta}} \frac{\lambda m}{2} \left\| \boldsymbol{\theta} - \boldsymbol{\theta}_t \right\|^2 + L(\boldsymbol{\theta}; x_t, y_t) + \text{constant}, \qquad (18)
\end{aligned}
$$

where in (i) we invoked the definition of convex conjugate; in (ii) we interchange $\min$ and $\max$ since strong duality holds (as stated in [Kakade and Shalev-Shwartz, 2008], a sufficient condition is that R is strongly convex, L is convex and $\operatorname{dom} L$ is polyhedral); and in (iii) we used the facts that $R(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|^2/2$ is conjugate of itself, and that $g(\mathbf{x}) = tf(\mathbf{x}-\mathbf{x}_0)$ implies $g^\star(\mathbf{y}) = \mathbf{x}_0^\top \mathbf{y} + tf^\star(\mathbf{y}/t)$.