Learning Structured Classifiers with Dual Coordinate Descent

André F. T. Martins*† Kevin Gimpel* Noah A. Smith* Eric P. Xing*
Pedro M. Q. Aguiar‡ Mário A. T. Figueiredo†

*School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA

{†Instituto de Telecomunicações, ‡Instituto de Sistemas e Robótica}
Instituto Superior Técnico
Lisboa, Portugal

{afm,kgimpel,nasmith,epxing}@cs.cmu.edu
aguiar@isr.ist.utl.pt, mtf@lx.it.pt

Abstract

We present a unified framework for online learning of structured classifiers. This framework handles a wide family of convex loss functions that includes as particular cases 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. No learning rate parameter is required. Our experiments show that the technique is faster to converge to an accurate model than stochastic gradient descent, on two NLP problems, at least when inference is exact.

1 Introduction

Many important problems in NLP, such as segmentation, parsing, and machine translation, require classifiers with structured outputs. Learning those 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 (LeCun et al., 1998; Vishwanathan et al., 2006), and the margin infused relaxed algorithm (MIRA, Crammer et al. 2006).

In this paper, we present 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 updates in the dual, generalizing the framework of Shalev-Shwartz and Singer (2006).

Although the derivation in §2–3 is rather technical, the update equations (§4) share the remarkable simplicity of stochastic gradient descent. Further, they do not require tuning a learning rate parameter. Instead, their learning rate is a function of the gradient of the loss and of the loss itself. We show how both quantities can be obtained by solving the variational problem. Well-known MIRA is a special case.

Two important problems in NLP provide an experimental testbed (§5): named entity recognition and dependency parsing. 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). As a by-product, we provide an interpretation of their (approximate) belief propagation algorithm by making explicit the underlying optimization problem (appendix A).

2 Structured Classification and Loss

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.

A **linear classifier** is a function $h_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$ defined as

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

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, $R(\boldsymbol{\theta}) \triangleq \frac{1}{2}\|\boldsymbol{\theta}\|^2$.

We focus on loss functions belonging to the family defined in Figure 1, which subsumes some well-known cases:

- The **log-loss** used in CRFs, denoted $L_{\mathrm{CRF}}(\boldsymbol{\theta}; x, y) \triangleq -\log P_{\boldsymbol{\theta}}(y|x)$, corresponds to $\beta = 1$ and $\gamma = 0$;
- The **hinge loss** of structured SVMs, $L_{\mathrm{SVM}}(\boldsymbol{\theta}; x, y) \triangleq \max_{y' \in \mathcal{Y}(x)} \boldsymbol{\theta}^\top(\boldsymbol{\phi}(x, y') - \boldsymbol{\phi}(x, y)) + \ell(y', y)$, corresponds to $\beta \to \infty$ 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 by 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 the optimization problem in Eq. 2, for this family of loss functions.

[1] Some important non-convex loss functions 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 $\beta = 1$ gives us $\delta L_{\beta,\gamma}(\boldsymbol{\theta}; x, y) = \log \mathbb{E}_{\boldsymbol{\theta}} \exp \ell(Y, y)$, which is an upper bound of the $\mathbb{E}_{\boldsymbol{\theta}} \ell(Y, y)$ function used in minimum risk training (Smith and Eisner, 2006). For $\beta = \infty$, $\delta L_{\beta,\gamma}$ becomes the *structured ramp loss* (Collobert et al., 2006; Chapelle et al., 2008).

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

Figure 1: A family of loss functions including as particular cases the ones used in CRFs, structured SVMs, and the structured perceptron. The hyperparameter β is the analogue of the inverse temperature in a Gibbs distribution, while γ scales the cost.

2.2 A Framework for Structured Inference

We now turn to inference problems, of which two important cases are: to obtain the most probable assignment (i.e., to solve the problem in Eq. 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.

We start by assuming that the elements of $\mathcal{Y}(x)$ can be naturally represented 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 depends on x.) We consider subsets $S \subseteq \{1, \ldots, I\}$ and write $\mathbf{y}_S = (y_i)_{i \in S}$, the vector of partial assignments. We assume a one-to-one map (not necessarily onto) from $\mathcal{Y}(x)$ to $\bar{\mathcal{Y}}$; we 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). This is a 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[2] feature vector $\boldsymbol{\phi}_C(x, \mathbf{y}_C)$ and define

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

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 Eq. 4 induces the

[2]Local with respect to the factor (although the features may depend on the entire input x).

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: Each i indexes a pair of words and $\mathcal{Y}_i$ is the set of possible arc labels plus a `NULL` symbol; 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.[3] 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, which creates loops (Smith and Eisner, 2008).

3 Variational Inference

3.1 Polytopes and Duality

Since in this paper we use convex duality, a brief description follows.[4] Denote by $\bar{\mathbb{R}} \triangleq \mathbb{R} \cup \{+\infty\}$ the extended reals. Given a function $f : \mathbb{R}^n \to \bar{\mathbb{R}}$, its **convex conjugate** is the function $f^\star : \mathbb{R}^n \to \bar{\mathbb{R}}$ defined as $f^\star(\mathbf{y}) = \sup_{\mathbf{x}} \mathbf{x}^\top \mathbf{y} - f(\mathbf{x})$. This concept is central in the variational representation of the log-partition function that we will show next.

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

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

[3] In phrase-structure parsing, each i indexes a pair of words and $\mathcal{Y}_i$ the set of constituent labels (including `NULL`) that can span that phrase.

[4] Boyd and Vandenberghe (2004) provide more detail.

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.[5] 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 $\boldsymbol{\theta}$).[6] The next proposition goes farther by linking the points of $\mathcal{Z}(x)$ to the distributions in $\mathcal{P}$. Below, we let $H(P_{\boldsymbol{\theta}}(.|x)) = -\sum_{y \in \mathcal{Y}(x)} P_{\boldsymbol{\theta}}(y|x) \log P_{\boldsymbol{\theta}}(y|x)$ denote the **entropy**, $\mathbb{E}_{\boldsymbol{\theta}}$ the expectation operator 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 $\mathbb{E}_{\boldsymbol{\theta}} \boldsymbol{\phi}(x, Y) = \mathbf{F}(x)\mathbf{z}$. 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 Eq. 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 most probable assignment $\hat{y} \triangleq \text{argmax}_{y \in \mathcal{Y}(x)} P_{\boldsymbol{\theta}}(y|x)$ 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

[5] The convex hull of $\{\mathbf{z}_1, \ldots, \mathbf{z}_k\}$ is the set of points that can be written in the form $\lambda_1 \mathbf{z}_1 + \ldots + \lambda_k \mathbf{z}_k$, where $\sum_{i=1}^k \lambda_i = 1$ and each $\lambda_i \geq 0$.

[6] Another way of defining the marginal polytope is as the set of factor marginals that are realizable by any distribution that factors according to the graph. Wainwright and Jordan (2008) consider log-linear models with a *canonical overcomplete parametrization*, i.e., whose sufficient statistics (features) at each factor are indicators of the configuration of that factor ($\chi_C(\mathbf{y}_C)$ in our notation).

Note also that different parameter vectors $\boldsymbol{\theta}$ may parametrize the *same* distribution $P_{\boldsymbol{\theta}}$; each equivalence class $[\boldsymbol{\theta}]$ is an affine subspace of $\mathbb{R}^d$.

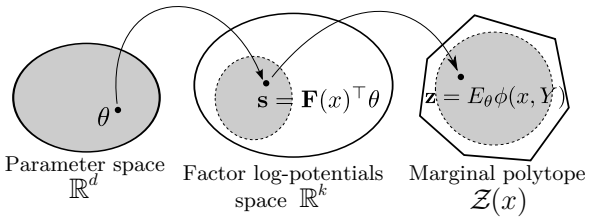

Figure 2: Dual parametrization of the distributions in $\mathcal{P}$. Our original parameter space (left) is first linearly mapped to the space of factor log-potentials (middle), called *canonical overcomplete parameter space* in Wainwright and Jordan (2008). The latter is mapped onto the relative interior of the marginal polytope $\mathcal{Z}(x)$ (right). In general only a subset of $\mathcal{Z}(x)$ is reachable from our parameter space. Any distribution in $\mathcal{P}$ can be parametrized by a vector $\boldsymbol{\theta} \in \mathbb{R}^d$ or by a point $\mathbf{z} \in \mathcal{Z}(x)$. See also footnote 6.

parameter space to the relative interior of the marginal polytope is surjective. The fact that the negative entropy is the convex conjugate of the log-partition function plays a key role in that proof. 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 $\boldsymbol{\theta} \mapsto \mathbf{s} = \mathbf{F}(x)^\top \boldsymbol{\theta}$ that "places" our parameters $\boldsymbol{\theta} \in \mathbb{R}^d$ into their canonical overcomplete parameter space. The image of this linear map is a linear subspace of the canonical overcomplete parameter space; in general our map $\boldsymbol{\theta} \mapsto \mathbf{z}$ is not onto $\mathrm{ri}\mathcal{Z}(x)$, unlike the case considered by Wainwright and Jordan (2008). Hence, our $H(\mathbf{z})$ is defined slightly differently: it can take the value $-\infty$ if no $\boldsymbol{\theta}$ maps to $\mathbf{z}$. However, this does not affect the expression in Eq. 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 $H(\mathbf{z})$. ■

Figure 2 provides an illustration of the dual parametrization implied by Prop. 1.

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 Fig. 1, and computing its gradient as a by-product.[7] This will be crucial for the learning algorithms to be introduced in §4.

[7]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."

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 $\mathbf{z}'$.[8] 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 Eq. 6 and after some algebra, we finally 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 Eq. 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.

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 modifying 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 $H(\bar{\mathbf{z}}) = \log Z(\boldsymbol{\theta}) - \boldsymbol{\theta}^\top \mathbf{F}(x)\bar{\mathbf{z}}$. The marginal polytope (call it $\mathcal{Z}_{\text{tree}}(x)$) is the same as the *arborescence polytope* of 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 Eq. 6, lack concise closed form expressions. Two approximate approaches have been recently proposed: a loopy belief propagation (BP) algorithm for computing pseudo-marginals, by Smith and Eisner (2008); and an LP-relaxation method for approximating the most likely parse tree, by Martins et al. (2009). Although the two methods may look unrelated at first sight, both are optimizing over outer bounds of the marginal polytope (although with distinct underlying factor graphs). Because we use Smith and Eisner's loopy BP approach in our experiments (§5), appendix A sheds some light on the matter by making explicit the variational approximation underlying that approach.

[8] 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.

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 Fig. 1. 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 log-loss, we obtain a new passive-aggressive algorithm for CRFs.

We start by noting that the learning problem in Eq. 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 carries over to the case where multiple epochs are allowed.

Proposition 2 *The Lagrange dual of the problem* $\min_{\boldsymbol{\theta}} P_t(\boldsymbol{\theta})$ *is*

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

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

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

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:* $\boldsymbol{\theta}^* = -\frac{1}{\lambda m} \sum_{i=1}^{t-1} \boldsymbol{\mu}_i^*$.

Proof: Adapted from Kakade and Shalev-Shwartz (2008). ■

We can see our goal as that of maximizing $D_{m+1}(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_m)$. Dual coordinate ascent (DCA) is an umbrella name for algorithms that make progress in the dual by manipulating 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 mapping of this subproblem back into the primal space is characterized by the next proposition, shedding light on the connections with known online algorithms.

Algorithm 1 Dual coordinate ascent (DCA)

Input: dataset $\mathcal{D}$, regularization coefficient λ, number of iterations K
Initialize $\boldsymbol{\theta}_1 \leftarrow \mathbf{0}$; 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 Eq. 11 exactly or approximately (see Alg. 2)
end for
Return the averaged model $\bar{\boldsymbol{\theta}} \leftarrow \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{\theta}_t$.

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

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

Proof: From Eq. 10, $\max_{\boldsymbol{\mu}} D_{t+1}(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{t-1}, \boldsymbol{\mu}) =$

$$\begin{aligned}
& \max_{\boldsymbol{\mu}} -\frac{1}{2\lambda m}\left\|\sum_{i=1}^{t-1}\boldsymbol{\mu}_i + \boldsymbol{\mu}\right\|^2 - L^\star(\boldsymbol{\mu}; x_t, y_t) - \sum_{i=1}^{t-1} L^\star(\boldsymbol{\mu}_i; x_i, y_i) \\
= & \max_{\boldsymbol{\mu}} -\frac{1}{2\lambda m}\|-\lambda m\boldsymbol{\theta}_t + \boldsymbol{\mu}\|^2 - L^\star(\boldsymbol{\mu}; x_t, y_t) + \text{const.} \\
= & \max_{\boldsymbol{\mu}} -\frac{1}{2\lambda m}\|-\lambda m\boldsymbol{\theta}_t + \boldsymbol{\mu}\|^2 - \max_{\boldsymbol{\theta}}(\boldsymbol{\mu}^\top\boldsymbol{\theta} - L(\boldsymbol{\theta}; x_t, y_t)) + \text{const.} \\
= & \min_{\boldsymbol{\theta}}\max_{\boldsymbol{\mu}} -\frac{1}{2\lambda m}\|-\lambda m\boldsymbol{\theta}_t + \boldsymbol{\mu}\|^2 - \boldsymbol{\mu}^\top\boldsymbol{\theta} + L(\boldsymbol{\theta}; x_t, y_t) + \text{const.} \\
= & \min_{\boldsymbol{\theta}}\left(\max_{\boldsymbol{\mu}} \boldsymbol{\mu}^\top(-\boldsymbol{\theta}) - \frac{1}{2\lambda m}\|\boldsymbol{\mu} - \lambda m\boldsymbol{\theta}_t\|^2\right) + L(\boldsymbol{\theta}; x_t, y_t) + \text{const.} \\
= & \min_{\boldsymbol{\theta}} \frac{\lambda m}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}_t\|^2 + L(\boldsymbol{\theta}; x_t, y_t) + \text{const.},
\end{aligned} \tag{12}$$

where we invoked the definition of convex conjugate, used the fact that strong duality holds, noted that the convex conjugate of $R(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|^2/2$ is itself, and that if $g(\mathbf{x}) = tf(\mathbf{x} - \mathbf{x}_0)$, then $g^*(\mathbf{y}) = \mathbf{x}_0^\top\mathbf{y} + tf^*(\mathbf{y}/t)$. ■

Assembling these pieces together yields Alg. 1, where the solution of Eq. 11 is carried out by Alg. 2, as explained next.[9] While the problem in Eq. 11 is easier than the batch problem in Eq. 2, an exact solution may still be prohibitively expensive

[9] Note the final averaging step in Alg. 1. This is a simple online-to-batch conversion scheme with good generalization guarantees; see Cesa-Bianchi et al. (2004) for details.

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 Eq. 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 $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \nabla L(\boldsymbol{\theta}_t; x_t, y_t)$

in large-scale settings, particulary 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 Eq. 11 as

$$\begin{array}{ll} \min\limits_{\boldsymbol{\theta},\xi} & \frac{\lambda m}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}_t\|^2 + \xi \\ \text{s.t.} & L(\boldsymbol{\theta}; x_t, y_t) \leq \xi, \quad \xi \geq 0. \end{array} \tag{13}$$

From the convexity of L, we may take its first-order Taylor approximation around $\boldsymbol{\theta}_t$ to obtain the lower bound $L(\boldsymbol{\theta}; x_t, y_t) \geq$

$$L(\boldsymbol{\theta}_t; x_t, y_t) + (\boldsymbol{\theta} - \boldsymbol{\theta}_t)^\top \nabla L(\boldsymbol{\theta}_t; x_t, y_t).$$

Therefore the true minimum in Eq. 11 is lower bounded by

$$\begin{array}{ll} \min\limits_{\boldsymbol{\theta},\xi} & \frac{\lambda m}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}_t\|^2 + \xi \\ \text{s.t.} & L(\boldsymbol{\theta}_t; x_t, y_t) + (\boldsymbol{\theta} - \boldsymbol{\theta}_t)^\top \nabla L(\boldsymbol{\theta}_t; x_t, y_t) \leq \xi, \\ & \xi \geq 0. \end{array} \tag{14}$$

This is a Euclidean projection problem (with slack) that admits the closed form solution

$$\boldsymbol{\theta}_{t+1} \quad = \quad \boldsymbol{\theta}_t - \eta_t \nabla L(\boldsymbol{\theta}_t; x_t, y_t)$$

with

$$\eta_t = \min\left\{\frac{1}{\lambda m}, \frac{L(\boldsymbol{\theta}_t; x_t, y_t)}{\|\nabla L(\boldsymbol{\theta}_t; x_t, y_t)\|^2}\right\}. \tag{15}$$

Example: 1-best MIRA If L is the hinge-loss, we obtain, from Eq. 9

$$\begin{array}{rcl} \nabla L_{\text{SVM}}(\boldsymbol{\theta}_t; x_t, y_t) & = & \mathbf{F}(x_t)(\bar{\mathbf{z}}_t - \mathbf{z}_t) \\ & = & \boldsymbol{\phi}(x_t, \hat{y}_t) - \boldsymbol{\phi}(x_t, y_t), \end{array}$$

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

$$\boldsymbol{\theta}_{t+1} \quad = \quad \boldsymbol{\theta}_t - \eta_t(\boldsymbol{\phi}(x_t, \hat{y}_t) - \boldsymbol{\phi}(x_t, y_t))$$

with

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

This is precisely the max-loss variant of the 1-best MIRA algorithm (Crammer et al., 2006). 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 log-loss. In that case, the exact problem in Eq. 13 can be expressed as

$$\begin{array}{ll} \min\limits_{\boldsymbol{\theta},\xi} & \frac{\lambda m}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}_t\|^2 + \xi \\ \text{s.t.} & -\log P_{\boldsymbol{\theta}}(y_t|x_t) \leq \xi, \quad \xi \geq 0. \end{array}$$

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 Eq. 9

$$\begin{aligned} \nabla L_{\text{CRF}}(\boldsymbol{\theta}_t; x_t, y_t) &= \mathbf{F}(x_t)(\bar{\mathbf{z}}_t - \mathbf{z}_t) \\ &= \mathbb{E}_{\boldsymbol{\theta}_t}\boldsymbol{\phi}(x_t, Y_t) - \boldsymbol{\phi}(x_t, y_t), \end{aligned}$$

where now $\bar{\mathbf{z}}_t$ is an expectation instead of a mode. The update becomes

$$\boldsymbol{\theta}_{t+1} \quad = \quad \boldsymbol{\theta}_t - \eta_t(\mathbb{E}_{\boldsymbol{\theta}_t}\boldsymbol{\phi}(x_t, Y_t) - \boldsymbol{\phi}(x_t, y_t))$$

with

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

Thus, the difference with respect to standard 1-best MIRA 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 loss 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 Eq. 11 can be built by using the variational representation machinery; see Eq. 8 for losses in the family $L_{\beta,\gamma}$. Plugging this variational representation into the constraint in Eq. 13 we obtain the following semi-infinite quadratic program:

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

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 Eq. 17 is a convex set defined by the intersection of uncountably many half-spaces (indexed by the points in the marginal polytope).[10] Our approximation consisted of relaxing the problem in Eq. 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.[11]

5 Experiments

We report experiments on two tasks: named entity recognition and dependency parsing. For each, we compare DCA (Alg. 1) with stochastic gradient descent (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 from LeCun et al. (1998), i.e., $\eta_t = \frac{\eta}{1+(t-1)/m}$. We choose η using dev-set validation after a single training iteration for each value of C (Collins et al., 2008).[12]

5.1 Named Entity Recognition

We use the English data from the CoNLL 2003 shared task (Tjong Kim Sang and De Meulder, 2003). The data consist of English news articles annotated with four

[10] 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.

[11] 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 $L_{\text{SVM}}(\boldsymbol{\theta}_t, x_t, y_t)$.

[12] One iteration is one pass over the training set.

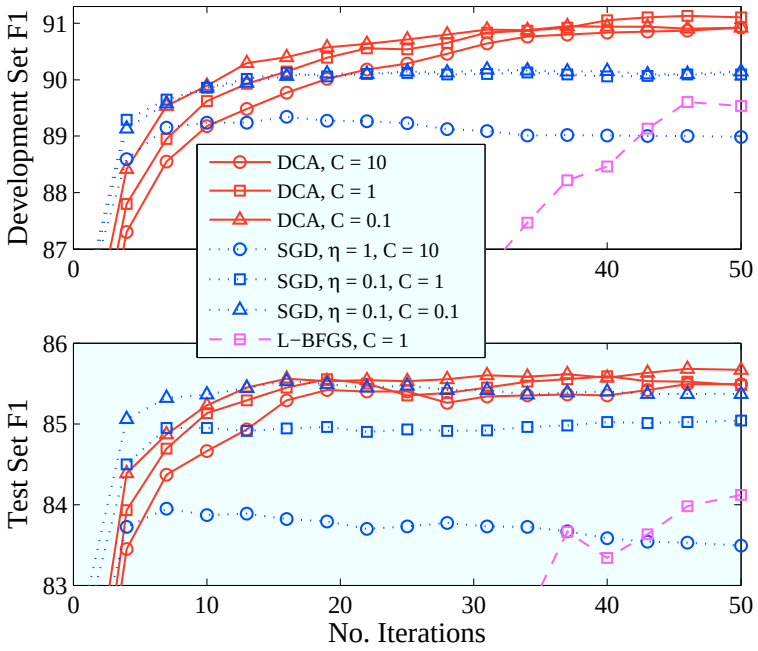

Figure 3: Named entity recognition. Learning curves for DCA (Alg. 1), SGD, and L-BFGS. The SGD curve for $C = 10$ is lower than the others because dev-set validation chose a suboptimal value of η; DCA, by contrast, does not require choosing any hyperparameters other than C. L-BFGS ultimately converges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

entity types: person, location, organization, and miscellaneous. We used a standard set of feature templates, following Kazama and Torisawa (2007) with token shape features like those in Collins (2002b) and simple gazetteer features; a feature was included iff it occurred at least once in the training data (total 1,312,255 features). The task is evaluated using the F_1 measure computed at the granularity of entire entities.

In addition to SGD, we also compare with L-BFGS (Liu and Nocedal, 1989), a frequently-used option for optimizing conditional log-likelihood (i.e., the CRF case). We used $\{0.001, 0.01, 0.1, 1, 10, 100\}$ for the set of η values considered for SGD. The results are shown in Figure 3. DCA reaches better-performing models than the baselines and only requires tuning a single hyperparameter.

5.2 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).[13] All experiments are evaluated using the *unlabeled attachment score* (UAS), with its default settings.[14] We adapted TurboParser[15] to handle any loss function $L_{\beta,\gamma}$ by implementing Alg. 1; for decoding, we implemented the loopy BP algorithm of Smith and Eisner (2008) (see §3.2).[16] We employed the pruning strategy described by Martins et al. (2009) and tried two different 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 Figure 4. For the arc-factored models, the learning curve of DCA seems to lead faster to an accurate model, with the additional advantage of not require tuning a learning rate parameter. For the second-order models of Danish and English, however, DCA did not perform so well.[17]

Finally, Table 1 shows results obtained for different settings of β and γ.[18] 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 eval-

[13] We used the provided train/test splits for all datasets. For training, sentences longer than 80 words were discarded. For testing, all sentences were kept.

[14] `http://nextens.uvt.nl/~conll/software.html`

[15] Publicly available at `http://www.ark.cs.cmu.edu/TurboParser`.

[16] We will release the modified code upon publication.

[17] Further analysis revealed that for about 15% of the training instances loopy BP led to very poor variational approximations of $\log Z(\boldsymbol{\theta})$ leading to estimates $P_{\boldsymbol{\theta}_t}(y_t|x_t) > 1$ and implying a negative value for the learning rate (see Eq. 16), that we truncate to zero. Thus, no update occurrs for those instances, which explains the somewhat slower convergence. There are many possibilities for fixing this problem: use better approximation techniques (or a combination of several), use techniques like the one in Wainwright and Jordan (2008) which guarantee upper bounds of the log-partition function, have a simple default strategy for the case where the estimate is obviously bad, etc. We defer this to future work.

[18] Observe that although three hyperparameters need to be tuned, only two degrees of freedom exist: indeed, the settings (λ, β, γ) and $(\lambda', \beta', \gamma')$ lead to an equivalent learning problem if $\lambda' = \lambda/a$, $\beta' = \beta/a$ and $\gamma' = a\gamma$ for any $a > 0$, with the corresponding solutions related via $\boldsymbol{\theta}' = a\boldsymbol{\theta}$.

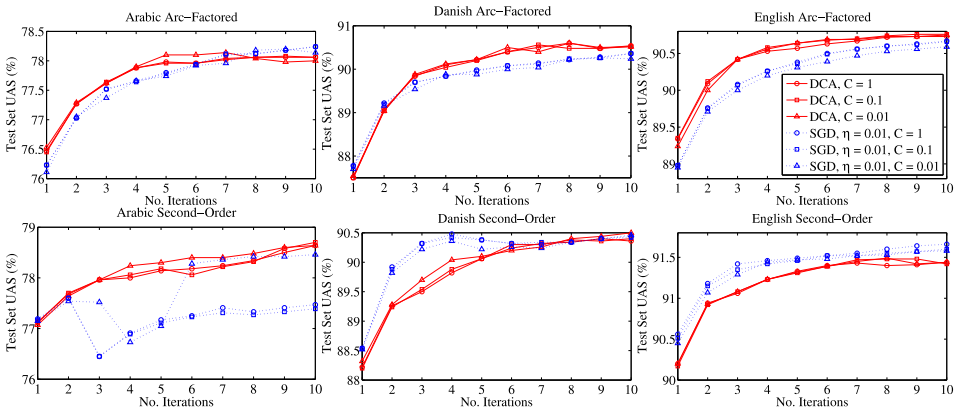

Figure 4: Dependency parsing. Learning curves for DCA (Alg. 1) and SGD, the latter with the learning rate $\eta = 0.01$ chosen from $\{0.001, 0.01, 0.1, 1\}$ using the same procedure as before. Although the two curves are very close for all languages and configurations, observe that with the exception of the second-order models for Danish and English, DCA seems to converge faster to an accurate model. *The difference would be higher had we shifted the SGD plots four iterations to the right to count the extra iterations for choosing the learning rate, a parameter which is absent in DCA.* The instability when training the second-order models might be due to the fact that inference there is approximate.

uating the loss function and its gradient. Experiments in named entity recognition and dependency parsing showed that the algorithm converges to accurate models at least as fast as stochastic gradient descent.

A Smith and Eisner's Loopy BP

We state explicitly the variational approximation that underlies the loopy BP approach of Smith and Eisner (2008), for a model with pairwise arc interactions (features on siblings and grandparents).

The factor graph for this model contains a `TREE` hard constraint factor, unary soft factors (one per candidate arc), and pairwise soft factors. Let A denote the set of candidate arcs and $P \subseteq A^2$ denote the set of pairs of arcs that have factors. All the variables are binary and the features only fire when an arc or pair of arcs are active; therefore, we may overload notation and write $\mathbf{z} = (\mathbf{z}_A, \mathbf{z}_P)$ with $\mathbf{z}_A = (z_a)_{a \in A}$ and $\mathbf{z}_P = (z_{ab})_{(a,b) \in P}$. Two approximations are made: first, $\mathcal{Z}(x)$ is replaced by a *local* polytope $\bar{\mathcal{Z}}(x) \supseteq \mathcal{Z}(x)$ specified by the constraints $z_{ab} \leq z_a$, $z_{ab} \leq z_b$ and $z_{ab} \geq z_a + z_b - 1$ for all $(a, b) \in P$, along with the tree constraint $\mathbf{z}_A \in \mathcal{Z}_{\text{tree}}(x)$. Second, the entropy $H(\mathbf{z})$ is replaced by the

	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	10.0	10.0	1.0
	F_1 (%)	85.48	85.54	85.65	85.72			85.41
DEPENDENCY	BEST C	0.1	0.01	0.01	0.01	0.01	0.01	0.1
PARSING	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 is 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 $K = 10$; for $\beta \to \infty$ (the SVM case), a different inference algorithm, the one in (Martins et al., 2009), was used. (This is due to numeric instabilities for large values of β when computing arc-marginals via the matrix-tree theorem.)

Bethe approximation

$$H_{\text{Bethe}}(\mathbf{z}) = H_{\text{tree}}(\mathbf{z}_A) - \textstyle\sum_{(a,b)\in P} I_{a;b}(\mathbf{z}), \tag{18}$$

where $H_{\text{tree}}(\mathbf{z}_A)$ is the entropy of the arc-factored distribution with potentials set to the pseudo-marginals $\mathbf{z}_A$, and $I_{a;b}(\mathbf{z})$ is the mutual information associated with each pairwise factor.[19] The approximate variational expression becomes $\log Z(\boldsymbol{\theta}) \approx$

$$\begin{array}{ll} \max_{\mathbf{z}} & \boldsymbol{\theta}^\top \mathbf{F}(x)\mathbf{z} + H_{\text{tree}}(\mathbf{z}_A) - \displaystyle\sum_{(a,b)\in P} I_{a;b}(\mathbf{z}) \\ \text{s.t.} & z_{ab} \le z_a, \quad z_{ab} \le z_b, \\ & z_{ab} \ge z_a + z_b - 1, \quad \forall (a,b) \in P, \\ & \mathbf{z}_A \in \mathcal{Z}_{\text{tree}}(x), \end{array} \tag{19}$$

and the maximizer corresponds to the pseudo-marginals returned by the loopy BP algorithm (if it converges).[20] This procedure allows us to obtain approximations of the loss $L_{\beta,\gamma}$ and its gradient (Eqs. 8–9).

[19] Namely, $I_{a;b}(\mathbf{z}) = \sum_{y_a,y_b} \tau_{ab}(y_a,y_b) \log \frac{\tau_{ab}(y_a,y_b)}{\tau_a(y_a)\tau_b(y_b)}$, where $\tau_{ab}(1,1) = z_{ab}$, $\tau_{ab}(1,0) = z_a - z_{ab}$, $\tau_{ab}(1,1) = z_b - z_{ab}$, $\tau_{ab}(0,0) = 1 - z_a - z_b + z_{ab}$, $\tau_a(1) = z_a$, $\tau_a(0) = 1 - z_a$, and similarly for $\tau_b(y_b)$. The reader familiar with Bethe approximations for pairwise graphs will note that Eq. 19 only differs from the pure pairwise case without hard constraints by replacing the sum of variable entropies, $\sum_{a\in A} H(\mathbf{z}_a)$ by the tree entropy $H_{\text{tree}}(\mathbf{z}_A)$.

[20] To be precise, any stationary point of BP is a local optimum of this variational problem (Yedidia et al., 2001). Multiple optima may exist: H_{Bethe} is not necessarily concave.

References

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

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

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

Censor, Y. and Zenios, S. A. (1997). *Parallel optimization: Theory, algorithms, and applications*. Oxford University Press.

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

Chapelle, O., Do, C., Le, Q., Smola, A., and Teo, C. (2008). Tighter bounds for structured estimation. In *NIPS*.

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

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

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

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

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2006). Online Passive-Aggressive Algorithms. *JMLR*, 7:551–585.

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

Gimpel, K. and Smith, N. A. (2010). Softmax-margin crfs: Training log-linear models with loss functions. In *Proc. of NAACL*.

Kakade, S. and Shalev-Shwartz, S. (2008). Mind the Duality Gap: Logarithmic regret algorithms for online optimization. In *NIPS*.

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

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

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

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

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

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

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

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

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

Povey, D., Kanevsky, D., Kingsbury, B., Ramabhadran, B., Saon, G., and Visweswariah, K. (2008). Boosted MMI for model and feature-space discriminative training. In *Proc. of ICASSP*.

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

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

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

Smith, D. A. and Smith, N. A. (2007). Probabilistic models of nonprojective dependency trees. In *Proc. EMNLP-CoNLL*.

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

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

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

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

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

Vishwanathan, S. V. N., Schraudolph, N. N., Schmidt, M. W., and Murphy, K. P. (2006). Accelerated training of conditional random fields with stochastic gradient methods. In *Proc. of ICML*.

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

Yedidia, J. S., Freeman, W. T., and Weiss, Y. (2001). Generalized belief propagation. In *NIPS*.