An Efficient Discriminative Training Method for Generative Models

Huizhen Yu
Dept. Computer Science
University of Helsinki
00014, Finland

Dimitri P. Bertsekas
Dept. EECS
Massachusetts Institute of Technology
Cambrige, 00239

Juho Rousu
Dept. Computer Science
University of Helsinki
00014, Finland

Abstract

We propose an efficient discriminative training method for generative models under supervised learning. In our setting, fully observed instances are given as training examples, together with a specification of variables of interest for prediction. We formulate the training as a convex programming problem, incorporating the SVM-type large margin constraints to favor parameters under which the maximum a posteriori (MAP) estimates of the prediction variables, conditioned on the rest, are close to their true values given in the training instances. The resulting optimization problem is, however, more complex than its quadratic programming (QP) counterpart resulting from the SVM-type training of conditional models, because of the presence of non-linear constraints on the parameters. We present an efficient optimization method, which combines several techniques, namely, a data-dependent reparametrization of dual variables, restricted simplicial decomposition, and the proximal point algorithm. Our method extends the one for solving the aforementioned QP counterpart, proposed earlier by some of the authors.

1 INTRODUCTION

We consider discriminative training of parameters for generative models with directed acyclic graphs and with discrete-valued variables. We assume a supervised learning setting, in which fully observed instances are given as training examples, together with a specification of variables of interest to prediction. These will be called hidden variables and they may vary from instance to instance. As to which variables are considered as hidden, it is sometimes naturally determined by the model or task, as in the case of a hidden Markov model (HMM) or a classification task. However, the selection can also be made only for the purpose of discriminative training: for example, in a way emulating the idea of the "coding technique" or "pseudo-likelihood" of Besag [Bes74], we can select a subset of nodes of a complex Bayesian network (BN) such that their edges cover many parts of the graph, and given the rest of the nodes, the inference on the graph is relatively easy.

There are certain circumstances where we may favor discriminative training over other statistics-based approaches, e.g., maximum likelihood (ML) parameter estimation: for example, when the number of instances is small, or when the instances are selected with biases and therefore do not reflect the true distribution, or when we want to adapt the parameters of a model learned in similar but different applications. In general, it may still be debatable whether discriminative training is preferred for generative models, as such training may introduce biases to the model. While our focus in this paper is primarily on the algorithmic aspects of efficient training, we emphasize here that assessing and preventing such biases are important issues, which we aim to address in the future.

We take the log-probabilities associated with the edges of the generative model as model parameters. We formulate the discriminative training as a convex programming problem. Its objective function has KL divergence terms to control the degree of deviation of the model parameters from certain given distributions, and penalty terms for the SVM-type margin violation to favor parameters under which the maximum a posteriori (MAP) estimates of the hidden variables, conditioned on the rest, are close to their true values given in the training instances. Large margin type of training criteria have been used for conditional models in recent structured prediction works (e.g., [Col02, ATH03, TGK04]); there, the resulting optimization problems are primarily convex quadratic

programming (QP) problems with a large number of linear constraints due to the margin penalty. In our context, the resulting optimization problems are more complex, because of the presence of non-linear constraints on the parameters (since probabilities have to sum to 1.)

We present an efficient optimization method; its idea applies to solving a class of problems resulting from the enforcement of large margin constraints. Our method is an extension of the one proposed earlier by two of the authors [YR07], which deals with the aforementioned QP problems resulting from non-kernelized formulations. It differs from the optimization methods of [TGK04, TJHA05, RSSS06, YR07] in its technique of handling the large number of margin constraints by a data-dependent linear reparametrization of dual variables. This technique reduces the dimension of the dual problem so that it is independent of the size of the prediction space, and is amenable to the use of efficient optimization methods of the feasible-direction type.[1] For problems with parameter constraints in addition to the margin constraints, a simple example has been demonstrated in [YR07], where the additional constraints are simple sign constraints. As the constraints considered here are more complex, we enhance these techniques by combining several ideas.

In broad terms, the method we propose in this paper operates at two levels. At the top level, we apply the proximal point algorithm and solve a sequence of regularized primal problems, which have nicer properties than the original problem and whose solutions converge to that of the latter. At the bottom level, we solve each regularized primal problem by dual optimization. In particular, by reparametrizing the multipliers associated with the large number of linear margin constraints, we derive an equivalent size-reduced dual problem which has an implicit polyhedral set constraint. We then use the restricted simplicial decomposition (RSD) method [HLV87] to deal with the set constraint, while we deal with the rest of the constraints directly.

The paper is organized as follows. In Sec. 2, we present the formulation of the training problem and our optimization algorithm. In Sec. 3, we describe two variants of the algorithm using subsets of the training set as working sets, including a variant with an online learning flavor, and we also describe extensions of our analysis to more general problem formulations. In Sec. 4, we present experimental results.

[1] A more detailed comparison to the early methods can be found in [YR07].

2 FORMULATION AND ALGORITHM

Let $\theta_i, i \in \mathcal{I}$ be vector-valued variables, each of which corresponds to a vector of log-conditional probabilities of some variable given its parents. Thus, $\theta_i \in \Re^{d_i}$ for some d_i and $\theta_i \leq 0$. Let $\mathcal{K}$ index the training examples, and for $k \in \mathcal{K}$, let $\mathcal{S}_k$ denote the space of all possible value assignments of the hidden variables in the k-th example.[2]

For each example instance, the log of the joint probability of hidden and non-hidden variables is a linear function of the log-probability parameters, with the coefficients being the respective counts of number of occurrences of certain patterns in the given instance, involving each variable and its parents. Let (s, o) denote the values of hidden and non-hidden variables, respectively, and let s^* be the values of hidden variables in the instance. The SVM-type large margin criterion is to require that after training, ideally, the true value s^* is the MAP estimate of the hidden variables and win by at least certain margin over other value assignments s:

$$\ln P(s, o) + l(s^*, s) \leq \ln P(s^*, o), \quad \forall\ s. \qquad (1)$$

Here $l(s^*, s)$ is a non-negative loss function with $l(s^*, s^*) = 0$ and may depend on the instance. It will be assumed that $l(\cdot, \cdot)$ has a favorable structure, so that the s maximizing the left-hand-side of Eq. (1),

$$\arg\max_s \ \big[\ln P(s, o) + l(s^*, s)\big],$$

can be computed efficiently. These problems are referred to as loss-augmented inference problems in the structured prediction literature. Solving them corresponds to identifying violating constraints in the convex program associated with the training, which is why their efficient solution methods are important for the training to be efficient. Expressing the margin constraints (1) in terms of parameters θ, we have linear inequalities the number of which equals the size of the space of s.

We define some shorthand notation. For a vector $x \in \Re^d$, $x = (x_1, x_2, \ldots, x_d)$, e^x denotes the vector $(e^{x_1}, e^{x_2}, \ldots, e^{x_d})$. Both vectors are treated as column vectors. The symbol $'$ denotes transpose. For $x \in \Re^d$, $\mathbf{1}'x$ will be used as a shorthand for $\sum_{j=1}^d x_j$, where we treat $\mathbf{1}$ as a vector of all ones with however a varying dimension depending on x. Let θ denote the collection

[2] In the case of HMM, for instance, θ_i is either a vector of log of state transition probabilities, or a vector of log of observation probabilities, corresponding to some state, and $\mathcal{S}_k$ is the space of the hidden state sequence for the k-th state and observation trajectory.

of variables $\theta_i, i \in \mathcal{I}$, and let ϵ denote the collection of scalar variables $\epsilon_k, k \in \mathcal{K}$.

2.1 PRIMAL PROBLEM

We formulate the training problem as solving the following convex program:

$$\text{(P)} \quad \min_{\theta,\epsilon} \quad -\sum_{i\in\mathcal{I}} c_i'\theta_i + \eta \sum_{k\in\mathcal{K}} \epsilon_k \tag{2}$$

$$\text{subj.} \sum_{i\in\mathcal{I}} a_{i,k}(s)'\theta_i + b_k(s) \le \epsilon_k, \quad \forall s \in \mathcal{S}_k, \ k \in \mathcal{K} \tag{3}$$

$$\mathbf{1}'e^{\theta_i} \le 1, \quad \forall i \in \mathcal{I} \tag{4}$$

$$\theta_i \le 0, \quad \forall i \in \mathcal{I}, \quad \epsilon_k \ge 0, \quad \forall k \in \mathcal{K} \tag{5}$$

The explanations for the constraints and the objective function are as follows.

The constraints in (3) correspond to the margin constraints of Eq. (1): the term $\sum_{i\in\mathcal{I}} a_{i,k}(s)'\theta_i$ corresponds to $\ln P(s,o) - \ln P(s^*,o)$, while the term $b_k(s)$ corresponds to the loss term $l_k(s^*,s)$.

The constraint $\mathbf{1}'e^{\theta_i} \le 1$ in (4) *does not enforce the sum of the associated probabilities to be* 1*, instead, it only requires the sum to be no greater than* 1. If we replace the inequality with the equality, the constraint would not be convex. However, apart from the pure convexity/algorithmic-related reason, we also have a natural interpretation for this constraint. We interpret the missing probability mass $1 - \mathbf{1}'e^{\theta_i}$, if $\mathbf{1}'e^{\theta_i} < 1$, to be the probability of the variable taking an "unknown" value, given its parents.[3] Such parameters do not cause practical problems for certain MAP inference, e.g., for predicting the hidden state sequence in HMM. (We observe in our experiments, most of e^{θ_i} do correspond to probability distributions, while the ones that do not usually have only a small amount of missing mass.

Each term in the summation $-\sum_{i\in\mathcal{I}} c_i'\theta_i$ in the objective function (2) comes from the KL-divergence $D(p \,\|\, q)$, defined for two d-dimensional probability distributions p and q by

$$D(p \,\|\, q) = \sum_{j=1}^{d} p_j \ln \frac{p_j}{q_j} = -H(p) - \sum_{j=1}^{d} p_j \ln q_j,$$

with $H(p)$ denoting the entropy of p. Here, for each i, we let $q = e^{\theta_i}$, while we let $p = c_i$ be some fixed distribution, which can be the maximum likelihood estimates of the parameters, or, for smoothing purposes, the uniform distribution, or, parameters estimated from a smaller model. Dropping the constant terms $H(c_i)$ then gives the term $-c_i'\theta_i$ in the objective. (Notice that $\sum_j q_j \le 1$ due to the constraint (4); but our use of the divergence formula is easily justified because we can think of p as having probability zero on the unknown value while q having $1 - \mathbf{1}'e^{\theta_i}$ on the latter.) Alternatively, one can choose other objective terms for θ, which are not necessarily linear. The optimization method to be presented shortly applies to more general cases, as will be discussed further in Sec. 3.

We note that because of the KL divergence terms $-\sum_{i\in\mathcal{I}} c_i'\theta_i$, the solution of (P) is non-degenerate, that is, the probabilities e^{θ_i} in optimal or near optimal solutions will not all tend to zero, as this will cause an infinite amount of increase in the objective.

2.2 PRELIMINARIES FOR THE ALGORITHM

The margin constraints (3) introduce potentially a huge number of linear constraints per example. To effectively deal with this, we will use the reparametrization approach proposed in [YR07]. Its idea is to reparametrize the associated multipliers by a *data-dependent linear transformation* that makes the dimension of the dual function independent of the size of assignment space $\mathcal{S}_k$. The reduced dual problem, which has an implicit polyhedral set constraint due to the reparametrization, is then optimized using feasible direction methods such as RSD. In this subsection, we first explain these ideas, (see [YR07] for more details), and we then point out a difficulty in applying RSD directly, which motivates our algorithm given in the next subsection.

2.2.1 Reparametrization

We derive our reparametrization essentially from the Lagrangian function. For each $k \in \mathcal{K}$, let $\beta_k(s), s \in \mathcal{S}_k$ be the multipliers associated with the constraints (3) and let β_k denote a collection of them. Let β denote the collection of $\beta_k, k \in \mathcal{K}$. Let λ_i be the multipliers associated with the constraints (4) for each $i \in \mathcal{I}$, and let λ denote the collection of $\lambda_i, i \in \mathcal{I}$. Define

$$\mu_i = \sum_{k\in\mathcal{K}, s\in\mathcal{S}_k} \beta_k(s) a_{i,k}(s), \quad \omega = \sum_{k\in\mathcal{K}, s\in\mathcal{S}_k} \beta_k(s) b_k(s). \tag{6}$$

[3] In the case of HMM, for instance, this will be either the probability of transition to an "unknown" state or the probability of generating an "unknown" observation at the state that θ_i is associated with.

We have the Lagrangian function

$$\begin{aligned} L\big((\theta,\epsilon),(\beta,\lambda)\big) &= -\sum_{i\in\mathcal{I}} c_i'\theta_i + \eta\sum_{k\in\mathcal{K}}\epsilon_k + \sum_{i\in\mathcal{I}}\mu_i'\theta_i + \omega \\ &\quad - \sum_{k\in\mathcal{K},s\in\mathcal{S}_k}\beta_k(s)\epsilon_k + \sum_{i\in\mathcal{I}}\lambda_i(\mathbf{1}'e^{\theta_i}-1) \\ &= \sum_{i\in\mathcal{I}}(\mu_i-c_i)'\theta_i + \omega + \sum_{i\in\mathcal{I}}\lambda_i(\mathbf{1}'e^{\theta_i}-1) \\ &\quad + \sum_{k\in\mathcal{K}}\Big(\eta - \sum_{s\in\mathcal{S}_k}\beta_k(s)\Big)\epsilon_k, \end{aligned}$$

from which, by minimizing over (θ,ϵ), we obtain the dual problem

$$\begin{aligned} \text{(D')} \qquad & \max_{\beta,\lambda}\ \omega - \sum_{i\in\mathcal{I}}\lambda_i + \sum_{i\in\mathcal{I}} q_i(\mu_i,\lambda_i) \\ & \text{subj. } \beta\ge 0,\quad \mathbf{1}'\beta_k\le\eta,\quad \forall k\in\mathcal{K} \\ & \qquad \lambda\ge 0,\quad (\mu,\omega) \text{ satisfy (6)} \end{aligned}$$

where

$$q_i(\mu_i,\lambda_i) = \min_{\theta_i\le 0}\ \Big[(\mu_i-c_i)'\theta_i + \lambda_i\mathbf{1}'e^{\theta_i}\Big]. \qquad (7)$$

Equivalently, we write the dual problem in terms of reparametrized variables (μ,ω,λ) with an implicit set constraint as

$$\begin{aligned} \text{(D)} \qquad & \max_{\mu,\omega,\lambda}\ \omega - \sum_{i\in\mathcal{I}}\lambda_i + \sum_{i\in\mathcal{I}} q_i(\mu_i,\lambda_i) \qquad (8) \\ & \text{subj. } \lambda\ge 0,\quad (\mu,\omega)\in\mathcal{D} \end{aligned}$$

where the set $\mathcal{D}$, determined by the parametrization in (6), is

$$\begin{aligned} \mathcal{D} = \Big\{(\mu,\omega)\ \Big|\ & \mu_i = \sum_{k\in\mathcal{K},s\in\mathcal{S}_k}\beta_k(s)a_{i,k}(s), \\ & \omega = \sum_{k\in\mathcal{K},s\in\mathcal{S}_k}\beta_k(s)b_k(s), \\ & \beta_k\ge 0,\quad \mathbf{1}'\beta_k\le\eta,\ \forall k\in\mathcal{K}\Big\}. \qquad (9) \end{aligned}$$

Notice that unlike β, the dimension of (μ,ω) is one plus the dimension of the parameters and does not depend on the size of $\mathcal{S}_k$. At an optimal dual solution $(\mu^*,\omega^*,\lambda^*)$, an optimal primal solution θ^* can be determined from (7).[4]

As the above reparametrization reduces the dimension of the dual function, it introduces a complicated implicit polyhedral set constraint $\mathcal{D}$. However, feasible direction methods can deal effectively with constraints of this type. There are also alternative reparametrizations based on the same idea, some of which will be given in Sec. 3.1.

[4]There is no duality gap and there exists at least one dual optimal solution, since the Slater constraint qualification holds for (P).

2.2.2 Dual Optimization with RSD

The method of simplicial decomposition [Hol74, Hoh77, HLV87] (see also [Ber99]) operates by solving iteratively two types of problems, called the *master problem* and the *direction-finding subproblem*, respectively. In the master problem, we optimize the objective function over an inner approximation of the feasible region, formed by the convex hull of a finite number of extreme points. This optimization can be done efficiently using the projected Newton method [Ber82], whenever the Hessian matrix can be conveniently computed. Subsequently, in the direction-finding subproblem, we find an extreme point along an ascent/descent direction, if the current solution is not a maximum/minimum. We form a new inner approximation of the feasible region by adding this point to the existing ones, thereby expanding the approximation into the part of the feasible region on which the objective can be improved. A new iteration of the master problem then starts.

The RSD algorithm [HLV87], in addition to following the above procedure, sets an upper limit on the number of extreme points forming the inner approximations,[5] and by doing so, it makes the complexity of solving master problems independent of that of the original problem. As the projected Newton method enjoys superlinear convergence rate (under certain standard conditions), master problems take relatively little time to solve, and the effectiveness of the algorithm thus depends on the effectiveness in making inner approximations of the feasible region. Therefore, there is an incentive not to restrict the number of points used in the inner approximation to be too small.

Let us outline the form of the RSD iterations in our context, under an ideal (but not correct) assumption that q_i were everywhere real-valued functions (the clarification on a domain related difficulty will be given shortly). It is only necessary to make inner approximations of $\mathcal{D}$, since the feasible region of λ is simple. Then, with $(\mu^j,\omega^j)\in\mathcal{D}, j=1,\ldots,m$ being m points whose convex hull makes an inner approximation of $\mathcal{D}$, the master problem takes the following equivalent form

$$\max_{\alpha,\lambda}\ \sum_{j=1}^m \alpha_j\,\omega^j - \sum_{i\in\mathcal{I}}\lambda_i + \sum_{i\in\mathcal{I}} q_i\Big(\sum_{j=1}^m \alpha_j\,\mu_i^j\ ,\ \lambda_i\Big) \qquad (10)$$

$$\text{subj. } \lambda\ge 0,\quad \alpha\ge 0,\quad \sum_{j=1}^m\alpha_j = 1 \qquad (11)$$

[5]When the number of points exceed the upper limit, we can simply discard some of them and add in the current feasible solution point.

The projected Newton method can be applied to solve this convex program with simple constraints [Ber82] (it operates by converting in a certain way the simplex constraint (11) to non-negativity constraints and then applying the Newton method with a crucial block diagonal condition on the scaling matrix). Corresponding to an optimum $(\bar{\alpha}, \bar{\lambda})$ of (10), the point $(\bar{\mu}, \bar{\omega}, \bar{\lambda})$ with $\bar{\mu} = \sum_{j=1}^{m} \bar{\alpha}_j \mu^j, \bar{\omega} = \sum_{j=1}^{m} \bar{\alpha}_j \omega^j$ is optimal for the master problem. Subsequently, with $Q(\mu, \omega, \lambda)$ denoting the dual function (8), the direction-finding subproblem aims to find a feasible point along an ascent direction by solving

$$\max_{(\mu,\omega)\in\mathcal{D}} \left[\nabla_\mu Q(\bar{\mu}, \bar{\omega}, \bar{\lambda})'(\mu - \bar{\mu}) + \nabla_\omega Q(\bar{\mu}, \bar{\omega}, \bar{\lambda})'(\omega - \bar{\omega}) \right].$$

As can be seen, the optimal value is zero if and only if $(\bar{\mu}, \bar{\omega}, \bar{\lambda})$ is optimal for the original problem. When the optimal value is nonzero, we use the optimal solution (μ, ω) to expand the inner approximation of $\mathcal{D}$. By the definition of Q and $\mathcal{D}$ [Eq. (9)] from our reparametrization, the direction finding subproblem is equivalent to

$$\max_{(\mu,\omega)\in\mathcal{D}} \left[\omega + \sum_{i\in\mathcal{I}} \nabla_{\mu_i} q_i(\bar{\mu}_i, \bar{\lambda}_i)' \mu_i \right]$$
$$= \sum_{k\in\mathcal{K}} \max_{s\in\mathcal{S}_k} \left[\sum_{i\in\mathcal{I}} \nabla_{\mu_i} q_i(\bar{\mu}_i, \bar{\lambda}_i)' a_{i,k}(s) + b_k(s) \right], \quad (12)$$

and the maximization problems in the expression on the right-hand-side have the same form as the loss-augmented inference problems and can be solved efficiently when there are favorable structures in the loss function and the generative models.

We make two observations. First, as the complexity of solving the master problems is unaffected by the training problem, we can exploit this property and save the number of calls to the inference algorithms by keeping a relatively high dimension of the simplex in the master problems. Second, due to the special structure of our problem, each term inside the summation in the right-hand-side of (12) is a feasible point in $\mathcal{D}$; whether it points to an ascent direction can be determined by calculating its inner product with the gradient. So, in our context, at the direction-finding step we can generate multiple ascent directions, instead of one, to expand the inner approximation.

2.2.3 A Difficulty Related to Domain

We cannot apply directly the above standard procedure of RSD, however, because the domain of the dual function (8) is not the entire space: the function q_i given in Eq. (7) is not everywhere real-valued, and its domain can be seen to be

$$\mathrm{dom}(q_i) = \{\mu_i \mid \mu_i \le c_i\} \times \Re. \quad (13)$$

Consequently, in order to start the RSD iterations, we need at least one point (μ^j, ω^j) in the master problem, Eq. (10), to be in the domain of the dual function, (otherwise, the value is $-\infty$); and furthermore, for the direction finding subproblems, we essentially need to find a point (μ, ω) along an ascent direction that lies not just in $\mathcal{D}$ but in the intersection of $\mathcal{D}$ and the Cartesian product of $\{\mu_i \mid \mu_i \le c_i\}$ over i and $\Re$. While finding such a point can be done, it is too computationally intensive, as far as our investigation shows.

To overcome this difficulty caused by the domain boundary of the dual function, we can add a small quadratic regularization term to the objective function of the primal problem, which then makes the corresponding dual function everywhere real-valued so that RSD as described above can be applied directly. Moreover, we do not have to settle for solving an approximation of the original problem: by changing the center of the quadratic term and solving a sequence of problems, we can approach the optimal solution of the original problem. This is the basis of the algorithm that we will present next, which is a proximal point algorithm from the primal viewpoint.

2.3 ALGORITHM

2.3.1 A Dual Proximal Point Algorithm

We apply the proximal point algorithm for solving the primal (P): at iteration n, we optimize (P_n), which differs from (P) only by a quadratic proximal term $\frac{\gamma_n}{2}\|\theta - \theta^n\|^2$ in the objective function:

$$(\mathrm{P}_n) \quad \min_{\theta,\epsilon} \quad -\sum_{i\in\mathcal{I}} c_i'\theta_i + \eta \sum_{k\in\mathcal{K}} \epsilon_k + \tfrac{\gamma_n}{2}\|\theta - \theta^n\|^2 \quad (14)$$
$$\text{subj.} \quad \sum_{i\in\mathcal{I}} a_{i,k}(s)'\theta_i + b_k(s) \le \epsilon_k, \ \forall s \in \mathcal{S}_k, \ k \in \mathcal{K} \quad (15)$$
$$\mathbf{1}' e^{\theta_i} \le 1, \quad \forall i \in \mathcal{I} \quad (16)$$
$$\epsilon_k \ge 0, \quad \forall k \in \mathcal{K} \quad (17)$$

Here, for all n, θ^n satisfies the constraints (16), with the initial θ^0 being any point, e.g., the log of the uniform distribution or the maximum likelihood estimates; γ_n is some small positive number and is bounded above for all n by some arbitrary positive number. We have dropped the constraints $\theta_i \le 0$ from (P), which are redundant anyway given the constraints (16), to simplify the dual problem. It can be seen that θ is an improved solution over θ^n for (P), if θ is for (P_n). Moreover, it is well known that if for all n, the center θ^{n+1} in the proximal term of (P_{n+1}) is the optimal solution of (P_n), then the sequence θ^n converges to an optimal solution of (P), assuming that

γ^n is bounded above (see e.g., [Ber99] and references given there). In our case, we will optimize (P_n) indirectly by optimizing its dual, and it turns out that the center θ^n can be changed in a more flexible way, as will be described in Sec. 2.3.3.

We consider solving (P_n) by dual optimization with the reparametrization and RSD introduced in Sec. 2.2. With the same parametrization in (6), we obtain the size-reduced dual problem of (P_n):

$$(D_n) \qquad \max_{\mu,\omega,\lambda} \ \omega - \sum_{i\in\mathcal{I}} \lambda_i + \sum_{i\in\mathcal{I}} q_i^n(\mu_i,\lambda_i) \qquad (18)$$
$$\text{subj. } \lambda \geq 0, \quad (\mu,\omega) \in \mathcal{D}$$

where $\mathcal{D}$ is as defined in Eq. (9), and the functions q_i^n are defined by [cf. the definition of q_i in Eq. (7)]:

$$q_i^n(\mu_i,\lambda_i) = \min_{\theta_i\in\Re^{d_i}} \Big[(\mu_i - c_i)'\theta_i + \lambda_i \mathbf{1}' e^{\theta_i} + \tfrac{\gamma_n}{2}\|\theta_i - \theta_i^n\|^2 \Big]. \qquad (19)$$

The latter are everywhere real-valued functions, and hence, so are the dual function Q_n of (D_n) [given in (18)]. The procedure of applying RSD is thus as described in Sec. 2.2 with Q_n replacing the dual function Q of (D), and with q_i^n replacing q_i in Eqs. (10) and (12).

Let us look more closely at how RSD operates with a varying sequence of convex programs instead of one. Since $\mathcal{D}$ is functionally independent of θ^n, inner approximations of $\mathcal{D}$ still are valid inner approximations, unaffected by θ^n. Similarly, a feasible solution $z_n = (\mu,\omega,\lambda)$ of (D_n) still is feasible for (D_{n+1}), and furthermore, since a small change in θ^n usually results in a small change in the dual optimal solution, z_n is a natural starting point for optimizing (D_{n+1}). Thus, while the actual function values and ascent directions are affected by a varying θ^n, they can be treated as generated from a black box, on top of which, RSD can run continuously without re-initializations for center changes (even though a careful re-initialization may give some speed-up). As a result, the overhead of the dual proximal point algorithm is not as high as might seem.

2.3.2 Details of Computation

The value, gradient and Hessian of the dual function Q_n of (D_n) are needed in RSD to optimize (D_n): the Hessian, or simply its diagonal entries, is useful for speeding up the convergence by applying the projected Newton method [Ber82]. We show how to compute these quantities efficiently. By the simple linear relations between q_i^n, Q_n and the objective function (10) of the master problem of RSD, we only need to show how to evaluate q_i^n and its first and second order derivatives efficiently.

Evaluation of function values: To compute $q_i^n(\bar\mu_i,\bar\lambda_i)$, we solve the minimization problem in (19), which has a unique minimum denoted by $\bar\theta_i$. Notice that this minimization problem is separable in each component of θ_i. For the j-th component, the necessary and sufficient optimality condition is

$$f(x) = \bar\mu_{ij} - c_{ij} - \gamma_n\,\theta_{ij}^n + \bar\lambda_i e^x + \gamma_n\, x = 0. \qquad (20)$$

The solution of Eq. (20) can be obtained by Newton's method:

$$x_{m+1} = x_m - \big(f'(x_m)\big)^{-1} f(x_m) = x_m - \frac{f(x_m)}{\bar\lambda_i e^{x_m} + \gamma_n},$$

which converges to $\bar\theta_{ij}$ globally with quadratic convergence rate; this follows from [OR70] (Theorems 13.3.4 and 13.3.7, p. 451-453) thanks to the convexity and other properties of f.

Evaluation of gradients: It can be seen that the gradients of $q_i^n(\bar\mu_i,\bar\lambda_i)$ are

$$\nabla_{\mu_i} q_i^n(\bar\mu_i,\bar\lambda_i) = \bar\theta_i, \qquad \nabla_{\lambda_i} q_i^n(\bar\mu_i,\bar\lambda_i) = \mathbf{1}' e^{\bar\theta_i}. \quad (21)$$

Evaluation of Hessians: The Hessians of $q_i^n(\bar\mu_i,\bar\lambda_i)$ can be derived as follows. Since $\bar\theta_{ij}$ satisfies Eq. (20), i.e.,

$$\bar\mu_{ij} - c_{ij} - \gamma_n\,\theta_{ij}^n + \bar\lambda_i e^{\bar\theta_{ij}} + \gamma_n\,\bar\theta_{ij} = 0,$$

by differentiating both sides with respect to μ_{ij} and λ_i, we have, respectively,

$$1 + \bar\lambda_i e^{\bar\theta_{ij}} \frac{\partial\bar\theta_{ij}}{\partial\mu_{ij}} + \gamma_n \frac{\partial\bar\theta_{ij}}{\partial\mu_{ij}} = 0$$
$$\Rightarrow \quad \frac{\partial\bar\theta_{ij}}{\partial\mu_{ij}} = \frac{-1}{\bar\lambda_i e^{\bar\theta_{ij}} + \gamma_n}, \qquad (22)$$
$$e^{\bar\theta_{ij}} + \bar\lambda_i e^{\bar\theta_{ij}} \frac{\partial\bar\theta_{ij}}{\partial\lambda_i} + \gamma_n \frac{\partial\bar\theta_{ij}}{\partial\lambda_i} = 0$$
$$\Rightarrow \quad \frac{\partial\bar\theta_{ij}}{\partial\lambda_i} = \frac{-e^{\bar\theta_{ij}}}{\bar\lambda_i + \gamma_n e^{\bar\theta_{ij}}}. \qquad (23)$$

Notice that $\frac{\partial\bar\theta_{ij}}{\partial\mu_{ij}}$ and $\frac{\partial\bar\theta_{ij}}{\partial\lambda_i}$ are smooth functions of μ_{ij} and λ_i, so, q_i^n and hence the dual function are twice differentiable. In particular, combining Eqs. (21)-(23), we see that the Hessian of q_i^n is given by

$$\frac{\partial^2 q_i^n(\bar\mu_i,\bar\lambda_i)}{\partial^2\mu_{ij}} = \frac{-1}{\bar\lambda_i e^{\bar\theta_{ij}} + \gamma_n}, \qquad (24)$$
$$\frac{\partial^2 q_i^n(\bar\mu_i,\bar\lambda_i)}{\partial\mu_{ij}\partial\lambda_i} = \frac{\partial^2 q_i^n(\mu_i,\lambda_i)}{\partial\lambda_i\partial\mu_{ij}} = \frac{-e^{\bar\theta_{ij}}}{\bar\lambda_i + \gamma_n e^{\bar\theta_{ij}}}, \qquad (25)$$
$$\frac{\partial^2 q_i^n(\bar\mu_i,\bar\lambda_i)}{\partial^2\lambda_i} = \sum_{j=1}^{d_i} \frac{-e^{2\bar\theta_{ij}}}{\bar\lambda_i + \gamma_n e^{\bar\theta_{ij}}}. \qquad (26)$$

2.3.3 Center-Change Rule and Bounds on Optimal Value

Let p_n^* and p^* denote the optimal value of (P_n) and (P), respectively. Let $P_n(\theta;\theta^n)$ and $P^o(\theta)$ be the function of θ obtained by minimizing the primal function of (P_n) and (P), respectively, over the slack variable ϵ.

Let $z_n = (\bar{\mu}, \bar{\omega}, \bar{\lambda})$ be the dual feasible solution at iteration n, which is the optimal solution of the most recent master problem. Let $\bar{\theta}^n$ be the vector whose components $\bar{\theta}_i^n$ solve the minimization problems defining $q_i^n(\bar{\mu}_i, \bar{\lambda}_i)$, respectively. The point $\bar{\theta}^n$ is not necessarily feasible for the primal problems, so we define $\bar{\theta}^{n,s}$ to be a primal feasible point obtained from $\bar{\theta}^n$ by a shift to each of the components,

$$\bar{\theta}_i^{n,s} = \bar{\theta}_i^n - \max\left\{0,\, \ln\left(\mathbf{1}' e^{\bar{\theta}_i^n}\right)\right\};$$

in other words, if $\mathbf{1}' e^{\bar{\theta}_i^n} > 1$, $e^{\bar{\theta}_i^{n,s}}$ is a scale of $e^{\bar{\theta}_i}$ by $\mathbf{1}' e^{\bar{\theta}_i^n}$. Then, since $p_n^* \le P_n(\bar{\theta}^{n,s};\theta^n)$ and $p^* \le P^o(\bar{\theta}^{n,s})$, we use the right-hand-sides to define upper bounds on p_n^* and p^*.

Lower bounds on p_n^* are obtained by the dual function values at iteration n. Lower bounds on p^* are not easy to compute. We make inaccurate estimations by evaluating the original dual function (D) – inaccurate, because not all $\bar{\mu}_i$ are necessarily in the domain of q_i, therefore, z_n is not necessarily in the domain of the dual function of (D). We project $\bar{\mu}_i$ to the domain of q_i (note that the projected point may not be in $\mathcal{D}$), and calculate the dual function value at the projected point. We also record how many $\bar{\mu}_i$ are outside the domain of q_i as a practical indicator for the reliability of the estimated lower bound.

In the standard/classical version of the proximal point algorithm, the center θ^n remains unchanged until problem (P_n) is fully solved. However, more efficient variants are often used, whereby θ^n changed to the current iterate once a suitable criterion is met. Simultaneously, γ_n may also be changed (usually decreased but could be also left constant). There is analysis corresponding to such variants, but this analysis must be modified for our situation because (P_n) is not solved directly. Instead the dual problem (D_n) is solved with the constraint set $\mathcal{D}$ being approximated by RSD. For our experiments, we have devised the following simple rule, and we will address the associated convergence issues in our extended report.

We set $\gamma_n = \gamma$ for all n. We fix some $\bar{\delta} > 0$ and a sequence of positive δ_m converging to 0. Let $m(n)$ be the number of center changes occurred before n. We set $\theta^{n+1} = \bar{\theta}^{n,s}$, if either (i) or (ii) of the following is true: (i) there is at least a $\bar{\delta}$ amount of improvement for (P): $p^o(\bar{\theta}^{n,s}) \le P^o(\theta^n) - \bar{\delta}$; (ii) the dual feasible solution z_n is $\delta_{m(n)}$-optimal for (D_n), in the sense that

$$\begin{aligned}
&\sup_{\lambda\ge 0,\ \lambda\in B(\bar{\lambda})} \Big[L\,\nabla_\lambda Q(\bar{\mu},\bar{\omega},\bar{\lambda})'(\lambda-\bar{\lambda})\Big] \\
&+ \sup_{(\mu,\omega)\in\mathcal{D}} \Big[\nabla_\mu Q(\bar{\mu},\bar{\omega},\bar{\lambda})'(\mu-\bar{\mu}) \\
&\qquad + \nabla_\omega Q(\bar{\mu},\bar{\omega},\bar{\lambda})'(\omega-\bar{\omega})\Big] \le \delta_{m(n)},
\end{aligned}$$

where L is some positive scalar and $B(\bar{\lambda})$ is the unit ball centered at $\bar{\lambda}$, (i.e., we check the violation of the optimality condition at z_n). It is evident that unless θ^n is optimal for (P) at some finite n, eventually only (ii) will be applied.

3 VARIANTS AND EXTENSIONS

3.1 TWO VARIANTS

While Sec. 2 presents the main idea of using reparametrization and simplicial decomposition, there are still many variations in the parametrization schemes and algorithms. Here we demonstrate two variants relating to partitioning the training set $\mathcal{K}$ into subsets, which we call working sets. Partitioning $\mathcal{K}$ into subsets $\mathcal{K}_j, j = 1,\dots,m$, which are used one at a time or in parallel fashion for optimization, can be especially helpful when the training set is large, and when the dimension of the parameters θ is high, but only a few of the parameters are typically relevant for each training example. The two cases usually happen simultaneously, as in natural language processing tasks, for instance. The first variant considers partitions with non-overlapping subsets, while the second variant deals with arbitrary subsets and works in an online learning setting.

A Coordinate Ascent Variant for Working Sets

The first variant is to apply, for each subset $\mathcal{K}_j$, a separate parametrization of the multipliers associated with $k \in \mathcal{K}_j$. To explain this, we write (μ,ω) as the sum of m vectors and correspondingly, $\mathcal{D}$ as the sum of m sets,

$$\mu_i = \sum_{j=1}^m \mu_i^j,\ \ i\in\mathcal{I},\ \ \omega = \sum_{j=1}^m \omega^j,\ \ \mathcal{D} = \mathcal{D}_1 \oplus \mathcal{D}_2 \cdots \oplus \mathcal{D}_m,$$

with $\mathcal{D}_j$ being the feasible region of (μ^j,ω^j) defined by, [cf. Eq. (9)],

$$\begin{aligned}
\mathcal{D}_j = \Big\{ (\mu^j,\omega^j) \,\Big|\, & \mu_i^j = \sum_{k\in\mathcal{K}_j, s\in\mathcal{S}_k} \beta_k(s) a_{i,k}(s), \\
& \omega^j = \sum_{k\in\mathcal{K}_j, s\in\mathcal{S}_k} \beta_k(s) b_k(s), \\
& \beta_k \ge 0,\ \ \mathbf{1}'\beta_k \le \eta,\ \forall k\in\mathcal{K}_j \Big\}. \qquad (27)
\end{aligned}$$

If training examples relating to similar subsets of parameters are grouped into common subsets, the effective dimension of each μ^j can be much lower than that of μ and θ, therefore, less memory and computation is needed for optimizing over μ^j alone.

We can apply a coordinate ascent type of optimization: optimize over a selected subset of variables $(\mu^j, \omega^j), j \in J$ and λ on the Cartesian product $(\prod_{j\in J} \mathcal{D}_j) \times \{\lambda \mid \lambda \geq 0\}$, while keeping the rest variables fixed. The application of RSD correspondingly takes a slightly different form. We maintain for each j a separate inner approximation $\widehat{\mathcal{D}}_j$ of $\mathcal{D}_j$. For the associated direction-finding subproblem, we only need to solve loss-augmented inference problems for the examples in $\mathcal{K}_j$. For the master problem, we optimize the dual function over the Cartesian product $(\prod_{j\in J} \widehat{\mathcal{D}}_j) \times \{\lambda \mid \lambda \geq 0\}$. The master problem thus has several simplex constraints instead of one, but can again be solved by applying the projected Newton method. It can be seen that the evaluation of function values and derivatives is the same as the one given in Sec. 2.3.

An Online Learning Variant for Working Sets

Given a stream of subsets $\mathcal{K}_t$ of $\mathcal{K}$ that covers $\mathcal{K}$ and can overlap with each other, our second variant of the algorithm uses $\mathcal{K}_t$ one at a time to optimize the dual function. However, it uses the reparametrization (μ, ω), without tracking the individual (μ^t, ω^t) as in the first variant.

To explain the idea, let us consider any subset $\mathcal{K}_j$ of $\mathcal{K}$ and its associated parametrization (μ^j, ω^j) and set $\mathcal{D}_j$ as given in Eq. (27). Notice a special property:

$$\mathcal{D}_j \subseteq \mathcal{D}, \quad \forall\, \mathcal{K}_j \subseteq \mathcal{K},$$

which holds because the origin is in all the sets $\mathcal{D}_j$, or in other words, we can set the rest of the multipliers $\beta_k(s)$ to zero for all $k \notin \mathcal{K}_j$ and $s \in \mathcal{S}_k$. Thus, given a subset $\mathcal{K}_t$ at time t, we can exploit the possibility of expanding the inner approximation of $\mathcal{D}$ into the part of $\mathcal{D}_j$ on which the dual function can be improved. In particular, we optimize the dual function over

$$\mathrm{conv}\big(\widehat{\mathcal{D}}_t^o \cup \widehat{\mathcal{D}}_t\big) \times \{\lambda \geq 0\},$$

where $\widehat{\mathcal{D}}_t^o$ is the inner approximation of $\mathcal{D}$ before the arrival of $\mathcal{K}_t$, and $\widehat{\mathcal{D}}_t$ is an inner approximation of $\mathcal{D}_t$, which keeps changing during the iteration t as we run RSD using the subset $\mathcal{K}_t$. At time $t+1$, we update $\widehat{\mathcal{D}}_t^o$ to $\widehat{\mathcal{D}}_{t+1}^o$ properly and proceed to the next subset $\mathcal{K}_{t+1}$. Like the effect of center-change discussed early, here, RSD can run in a way "indifferent" to the subset change, with the generation of feasible directions being treated as a black box.

For an offline learning setting, this sequential use of $\mathcal{K}_t$ may not be as efficient as the first variant, and furthermore, if $\mathcal{K}_t$ are not chosen properly, the solution may be confined in a subset of the feasible region. Nevertheless, the second variant is convenient to apply, and the ideas of the two variants can be combined.

3.2 MORE GENERAL OBJECTIVES AND CONSTRAINTS

The method and analysis of Sec. 2 for solving (P) are generally applicable to solving large-margin type of training problems of the following form,

$$\min_{\theta,\epsilon\geq 0} \quad \sum_{i\in\mathcal{I}} f_i(\theta_i) + \eta \sum_{k\in\mathcal{K}} h(\epsilon_k) \tag{28}$$

$$\text{subj.} \quad A_k(s)'\theta + b_k(s) \leq \epsilon_k, \quad \forall s \in \mathcal{S}_k, \quad k \in \mathcal{K} \tag{29}$$

$$g_{ij}(\theta_i) \leq 0, \quad j = 1, \ldots, m, \quad i \in \mathcal{I} \tag{30}$$

where Eq. (29) are margin constraints, Eq. (30) are convex parameter constraints independent of the data, f_i and h are convex functions, with h being a penalty function for margin violation.

As an example of f_i different to the one used in (P), we can consider again the KL-divergence $D(p \,\|\, q)$, but associate p with the parameters and q with some fixed distribution. One can show that the algorithm of Sec. 2.3 applies to solving the convex program with the new KL divergence terms replacing those in (P).

The penalty function h for margin violation affects the reparametrization and the set $\mathcal{D}$: sometimes variables in addition to (μ, ω) are introduced to simplify the dual problem, and the application of RSD can also be slightly different. It is straightforward to work out these details for specific choices of h, similar to Sec. 2.2; illustrations for certain cases (loss-rescaled slacks and quadratic penalties) can be found in [YR07].

4 PRELIMINARY EXPERIMENTS

We report preliminary experimental results on two data sets. One is artificially created, and the other is the Connect data set from the UCI machine learning repository. Additional experimental results will be reported in an extended version of this paper.

Our first experiment is on a toy tracking problem with an HMM model. There are 10 states and 7 observations. This results in 21 log-probability parameter vectors with a total dimension of 180. The 10 states correspond to 10 equally spaced positions on a ring, and the transition matrix is generated by perturbing the deterministic transition along the ring with a small probability (≈ 0.3) of transitioning to a random state, while the initial distribution is uniform. States 1 to 7

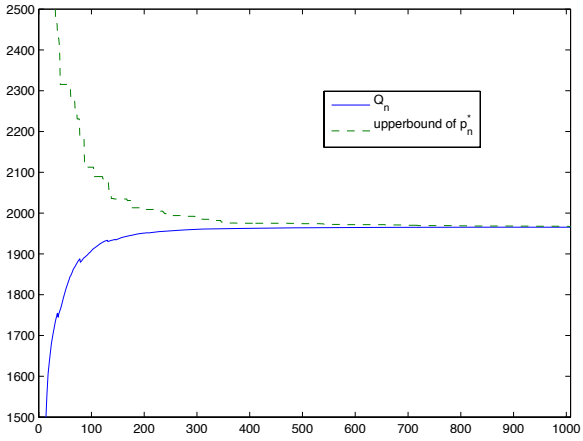

Figure 1: Behavior of the Algorithm on a Toy Tracking Problem with HMM.

tend to generate observations 1 to 7, respectively, but they also have a small probability of generating a random observation. For the last 3 states, the observation distribution is uniform.

We generate 100 training sequences and 1000 testing sequences all of length 50. For the divergence terms in the objective function, we choose all c_i to be uniform distributions. We compare the discriminatively trained model (DT) with that estimated by ML, on their MAP state estimation performance. When we use the 0/1 Hamming loss, we find that DT performs nearly as well as ML, but cannot outperform it. When we set the loss function to be the distance between the predicted position and the true one measured along the ring, DT noticeably outperforms ML. In particular, on the test data, DT has an average loss 94.6 per sequence with standard deviation 13.5, while ML has an average loss 103.4 with standard deviation 19.1. We also test the Hamming loss of the DT trained with the above loss function; it performs nearly as well as ML, making, on average, 2 more mistakes per sequence than ML.

Our algorithm, initialized with θ^0 corresponding to uniform distributions, solves the optimization problem (to within 0.1%-optimal) in about 1000 iterations (see Fig. 1). (Here, an iteration refers to an RSD iteration.) It runs in MATLAB. Table 1 shows the total cpu time in seconds and the percentage of it spent on solving master problems, direction finding, and handling center changes, respectively. The maximal dimension of the simplex in RSD is set to be 100; at the final solution, the dimension of the simplex is 61.

Our second experiment is performed on the UCI data set Connect, with a BN structure learned by a structure learning software independent of our algorithm. There are 43 nodes in the graph, one of which is a class

	Total	Master	Dir. Finding	CC
T	717s	3.4%	86.6%	10.0%
C	4627s	7.2%	48.5%	44.3%

Table 1: Time Usage of Optimization Subroutines: (T) refers to the tracking problem; (C) refers to the Connect data set; and the (CC) column refers to direction re-finding after an immediate center change.

variable taking 3 values. We take the subgraph around the class variable, which has 13 nodes and covers 172 log-probability parameter vectors with their total dimension being 513. There are over $60,000$ instances in this data set, among which we randomly select $30,000$ for training and keep the rest for testing. We further split the training set evenly into 60 subsets, and apply the online learning variant of the algorithm given in Sec. 3.1. We choose c_i in the objective function to be uniform distributions, and we initialize the algorithm with θ^0 corresponding to uniform distributions. Since the data set is too large, we only run the algorithm for 6 rounds over the 60 subsets sequentially; during each round, for each subset, we run maximally 3 RSD iterations. The total cpu time and the percentages spent on solving various subproblems are given in Table 1. We have not optimized the criterion for changing center. We compare DT with ML estimated on the entire data set. On both training and testing sets, DT makes about 100 less mistakes in predicting the class than ML. (The exact numbers for DT are $6,939/8,766$ mistakes on training/testing sets, respectively, while those for ML $7,023/8,872$.)

5 DISCUSSION

We have presented an optimization method for efficient discriminative training of generative models. Our technique applies to a class of problems associated with large margin type of training. Besides algorithmic improvements, future work may lead to better understanding of the effect on the solution of various aspects of the problem formulation, including the relaxed constraint (4), and the tradeoff between faithfulness to the data and discriminative capacity.

References

[ATH03] Y. Altun, I. Tsochantaridis, and T. Hofmann, *Hidden Markov support vector machines*, Proc. 20th Int. Conf. Machine Learning, 2003.

[Ber82] D. P. Bertsekas, *Projected Newton methods for optimization problems with simple con-*

straints, SIAM J. Control and Optimization **20** (1982), no. 2, 221–246.

[Ber99] ———, *Nonlinear programming*, 2nd ed., Athena Scientific, Belmont, MA, 1999.

[Bes74] J. Besag, *Spatial interaction and the statistical analysis of lattice systems*, J. Royal Statistical Society, Series B **36** (1974), no. 2, 192–236.

[Col02] M. Collins, *Discriminative training methods for hidden Markov models*, Proc. Conf. Empirical Methods in Natural Language Processing, 2002.

[HLV87] D. W. Hearn, S. Lawphongpanich, and J. A. Ventura, *Restricted simplicial decomposition: Computation and extensions*, Mathematical Programming Study **31** (1987), 99–118.

[Hoh77] B. Von Hohenbalken, *Simplicial decomposition in nonlinear programming algorithms*, Mathematical Programming **13** (1977), 49–68.

[Hol74] C. A. Holloway, *An extension of the Frank-Wolfe method of feasible directions*, Mathematical Programming **6** (1974), 14–27.

[OR70] J. M. Ortega and W. C. Rheinboldt, *Iterative solution of nonlinear equations in several variables*, Academic Press, New York, 1970.

[RSSS06] J. Rousu, C. Saunders, S. Szedmak, and J. Shawe-Taylor, *Kernel-based learning of hierarchical multilabel classification models*, J. Machine Learning Research **7** (2006), 1601–1626.

[TGK04] B. Taskar, C. Guestrin, and D. Koller, *Max-margin Markov networks*, Proc. Advances in Neural Information Processing Systems 16, 2004.

[TJHA05] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, *Large margin methods for structured and interdependent output variables*, J. Machine Learning Research **6** (2005), 1453–1484.

[YR07] H. Yu and J. Rousu, *An efficient method for large margin parameter optimization in structured prediction problems*, Technical Report C-2007-87, University of Helsinki, 2007.