Exponentiated Gradient Algorithms for Log-Linear Structured Prediction

Amir Globerson GAMIR@CSAIL.MIT.EDU
Terry Y. Koo MAESTRO@CSAIL.MIT.EDU
Xavier Carreras CARRERAS@CSAIL.MIT.EDU
Michael Collins MCOLLINS@CSAIL.MIT.EDU
Computer Science and Artificial Intelligence Laboratory, MIT, Cambridge MA 02139 USA

Abstract

Conditional log-linear models are a commonly used method for structured prediction. Efficient learning of parameters in these models is therefore an important problem. This paper describes an exponentiated gradient (EG) algorithm for training such models. EG is applied to the convex dual of the maximum likelihood objective; this results in both sequential and parallel update algorithms, where in the sequential algorithm parameters are updated in an *online* fashion. We provide a convergence proof for both algorithms. Our analysis also simplifies previous results on EG for max-margin models, and leads to a tighter bound on convergence rates. Experiments on a large-scale parsing task show that the proposed algorithm converges much faster than conjugate-gradient and L-BFGS approaches both in terms of optimization objective and test error.

1. Introduction

Structured learning problems involve the prediction of objects such as sequences or parse trees. The set of possible objects may be exponentially large, but each object typically has rich internal structure. Several models that implement learning in this scenario have been proposed over the last few years (Taskar et al., 2003; Lafferty et al., 2001). The underlying idea in these methods is to classify each instance $x \in \mathcal{X}$ into a label $y \in \mathcal{Y}$ using a prediction rule $y = \arg\max_{\hat{y}} \mathbf{w} \cdot \boldsymbol{\phi}(x, \hat{y})$. Here $\boldsymbol{\phi}(x, y)$ is a feature vector, and $\mathbf{w}$ is a set of weights that are learned from labeled data. The methods differ in their approach to learning the weight vector. Maximum margin Markov networks (Taskar et al., 2003) rely on margin maximization, while conditional random fields (CRFs) (Lafferty et al., 2001) construct a conditional log-linear model and maximize the likelihood of the observed data. Empirically both methods have shown good results on complex structured-prediction tasks.

In both CRFs and max-margin models, the learning task involves an optimization problem which is convex, and therefore does not suffer from problems with local minima. However, finding the optimal vector $\mathbf{w}$ may still be computationally intensive, especially for very large data sets. In the current paper, we propose a fast and efficient algorithm for optimizing log-linear models such as CRFs.

To highlight the difficulty we address, consider the common practice of optimizing the conditional log-likelihood of a CRF using conjugate-gradient or L-BFGS algorithms (Sha & Pereira, 2003). Any update to the weight vector would require at least one pass over the data set, and typically more due to line-search calls. Since conjugate gradient convergence typically requires several tens of iterations if not hundreds, the above optimization scheme can turn out to be quite slow. It would seem advantageous to have an algorithm that updates the weight vector after every sample point, instead of after the entire data set. One example of such algorithms is stochastic gradient descent and its variants (Vishwanathan et al., 2006).

A different approach, which we employ here, was first suggested by Jaakkola and Haussler (1999). It proceeds via the convex dual problem of the likelihood maximization problem. As in the dual Support Vector Machine (SVM) problem, the dual variables correspond to data points x_i. Specifically, to every point x_i there corresponds a distribution $\alpha_{i,y}$ (i.e. $\alpha_{i,y} \geq 0$ and $\sum_y \alpha_{i,y} = 1$). It is thus tempting to optimize $\alpha_{i,y}$ for each i separately, much like the Sequential Minimization Optimization (SMO) algorithm (Platt,

Appearing in *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, OR, 2007. Copyright 2007 by the author(s)/owner(s).

1998) often used to optimize SVMs. Several authors (e.g. Keerthi et al. 2005) have studied such an approach, using different optimization schemes, and ways of maintaining the distribution constraints on $\alpha_{i,y}$.

Here we present an algorithm based on exponentiated gradient (EG) updates (Kivinen & Warmuth, 1997).[1] These updates are specifically designed for optimizing over distributions, and were recently used in the max-margin setting (Bartlett et al., 2004). Our use of EG results in very simple updates that can be performed for every sample point, giving an online-like algorithm. Despite its online nature, our algorithm is guaranteed to improve the dual objective at each step, and this objective may be calculated after every single example without performing a pass over the entire data set. In this sense the algorithm is different from stochastic gradient descent.

We provide a convergence proof of the algorithm, using a symmetrization of the KL divergence. Our proof is relatively simple, and the convergence rate analysis applies both to the max-margin and log-linear settings. Furthermore, our proof for the max-margin case yields a larger learning rate than that used in the proof of Bartlett et al. (2004), giving a tighter bound on the rate of convergence. More generally, our convergence results may have relevance to optimization of general convex functions over the simplex using EG.

Finally, we compare our EG algorithm to conjugate gradient and L-BFGS algorithms on a standard multiclass learning problem, and a complex natural language parsing problem. In both settings we show that EG converges much faster than these algorithms, both in terms of objective function and classification error.

2. Primal and Dual Problems for Log-Linear Models

Consider a supervised learning setting with objects $x \in \mathcal{X}$ and labels $y \in \mathcal{Y}$.[2] In the structured learning setting, the labels may be sequences, trees, or other high-dimensional data with internal structure. Assume we are given a function $\boldsymbol{\phi}(x,y) : \mathcal{X} \otimes \mathcal{Y} \rightarrow \Re^d$ that maps (x,y) pairs to feature vectors. Given a parameter vector $\mathbf{w} \in \Re^d$, a conditional log-linear model defines a distribution over labels as

$$p(y|x;\mathbf{w}) = \frac{1}{Z_x} e^{\mathbf{w}\cdot\boldsymbol{\phi}(x,y)}$$

where Z_x is the partition function. Classification for x is then done by finding $y^* = \arg\max_y \mathbf{w}\cdot\boldsymbol{\phi}(x,y)$.

We turn to the problem of learning $\mathbf{w}$ from labeled data. Given a training set $\{(x_i,y_i)\}_{i=1}^n$, we wish to find $\mathbf{w}$ which maximizes the regularized log-likelihood

$$\underline{\text{P-LL}}: \quad \mathbf{w}^* = \arg\max_{\mathbf{w}} \textstyle\sum_i \log p(y_i|x_i;\mathbf{w}) - \frac{C}{2}\|\mathbf{w}\|^2$$

Here $C > 0$ is a constant determining the amount of regularization. The above is a convex optimization problem, and has an equivalent convex dual which was derived by Lebanon and Lafferty (2001). Denote the dual variables by $\alpha_{i,y}$ where $i = 1,\ldots,n$ and $y \in \mathcal{Y}$. We also use $\boldsymbol{\alpha}$ to denote the set of all variables, and $\boldsymbol{\alpha}_i$ the set of all variables corresponding to a given i. Thus $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_1,\ldots,\boldsymbol{\alpha}_n]$. Define the function $Q(\boldsymbol{\alpha})$ as

$$Q(\boldsymbol{\alpha}) = \sum_i \sum_y \alpha_{i,y}\log\alpha_{i,y} + \frac{1}{2C}\|\mathbf{w}(\boldsymbol{\alpha})\|^2$$

where

$$\mathbf{w}(\boldsymbol{\alpha}) = \sum_i \sum_y \alpha_{i,y}\boldsymbol{\psi}_{i,y} \tag{1}$$

and where $\boldsymbol{\psi}_{i,y} = \boldsymbol{\phi}(x_i,y_i) - \boldsymbol{\phi}(x_i,y)$. We shall find the following matrix notation convenient:

$$Q(\boldsymbol{\alpha}) = \sum_i \sum_y \alpha_{i,y}\log\alpha_{i,y} + \frac{1}{2}\boldsymbol{\alpha}^T A\boldsymbol{\alpha} \tag{2}$$

where A is a matrix of size $n|\mathcal{Y}| \times n|\mathcal{Y}|$ indexed by pairs (i,y) and $A_{(i,y),(j,z)} = \frac{1}{C}\boldsymbol{\psi}_{i,y}\cdot\boldsymbol{\psi}_{j,z}$.

In what follows we denote the set of distributions over $\mathcal{Y}$, i.e. the $|\mathcal{Y}|$-dimensional probability simplex, by Δ. The Cartesian product of n distributions over $\mathcal{Y}$ will be denoted by Δ^n. The dual optimization problem is then

$$\underline{\text{D-LL}}: \quad \boldsymbol{\alpha}^* = \begin{array}{ll} \arg\min & Q(\boldsymbol{\alpha}) \\ s.t. & \boldsymbol{\alpha} \in \Delta^n \end{array} \tag{3}$$

The duality between P-LL and D-LL implies that the primal and dual solutions satisfy $C\mathbf{w}^* = \mathbf{w}(\boldsymbol{\alpha}^*)$.

3. Exponentiated Gradient Algorithms

In this section we describe batch and online algorithms for solving D-LL . The algorithms we describe are based on Exponentiated Gradient updates, originally introduced by Kivinen and Warmuth (1997) in

[1] Note that EG has also been studied in the optimization literature, where it is known as Mirror Descent (e.g., see Beck and Teboulle (2003) for recent results).

[2] In reality the set of labels for a given example x may be a set $\mathcal{Y}(x)$ that depends on x. For simplicity, in this paper we use notation where $\mathcal{Y}$ is fixed for all x; it is straightforward to extend our notation to the more general case. In fact, in our experiments on dependency parsing the set $\mathcal{Y}$ does depend on x.

the context of online learning algorithms. More recently, Bartlett et al. (2004) have applied EG to a max-margin structured learning task, using it to minimize the dual of the max-margin formulation.

The EG updates rely on the following operation. Given a set of distributions $\boldsymbol{\alpha} \in \Delta^n$, a new set $\boldsymbol{\alpha}'$ can be obtained as

$$\alpha'_{i,y} = \frac{1}{Z_i}\alpha_{i,y}e^{-\eta\nabla_{i,y}}$$

where $\nabla_{i,y} = \frac{\partial Q(\boldsymbol{\alpha})}{\partial \alpha_{i,y}}$, $Z_i = \sum_{\hat{y}} \alpha_{i,\hat{y}}e^{-\eta\nabla_{i,\hat{y}}}$ and the parameter $\eta > 0$ is a learning rate.

We will also use the notation $\alpha'_{i,y} \propto \alpha_{i,y}e^{-\eta\nabla_{i,y}}$ where the partition function should be clear from the context.

Clearly $\boldsymbol{\alpha}' \in \Delta^n$ by construction. For the dual log-linear function $Q(\boldsymbol{\alpha})$ the gradient is

$$\nabla_{i,y} = 1 + \log \alpha_{i,y} + \frac{1}{C}\mathbf{w}(\boldsymbol{\alpha}) \cdot \boldsymbol{\psi}_{i,y}$$

The EG algorithm generates a sequence of distributions $\boldsymbol{\alpha}^t$ for $t = 1, \ldots, (T+1)$ by applying the above update repeatedly. We can consider two approaches:

- **Batch**: At every iteration the $\boldsymbol{\alpha}_i^t$ are simultaneously updated for all $i = 1, \ldots, n$.
- **Online**: At each iteration a single i is chosen and $\boldsymbol{\alpha}_i^t$ is updated to give $\boldsymbol{\alpha}_i^{t+1}$. We focus on the case where the values for i are chosen in a cyclic order.

Pseudo-code for both schemes is given in Figures 1 and 2. From here on we will refer to the batch and online EG algorithms applied to the log-linear dual as LLEG-Batch and LLEG-Online respectively.

In the next section we give convergence proofs for the LLEG-Batch and LLEG-Online algorithms. The techniques used in the convergence proofs are quite general, and could potentially be useful in deriving EG algorithms for other convex functions Q.

4. Convergence Results

4.1. Divergence Measures

Before providing convergence proofs, we define several divergence measures between distributions. Define the KL divergence between two distributions $\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i \in \Delta$ to be $D[\boldsymbol{\alpha}_i \| \boldsymbol{\beta}_i] = \sum_y \alpha_{i,y} \log \frac{\alpha_{i,y}}{\beta_{i,y}}$. Given two sets of n distributions $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \Delta^n$ define their KL divergence as $D[\boldsymbol{\alpha} \| \boldsymbol{\beta}] = \sum_i D[\boldsymbol{\alpha}_i \| \boldsymbol{\beta}_i]$.

```
Inputs: Examples $\{(x_i, y_i)\}_{i=1}^n$, learning rate $\eta > 0$.
Initialization: Set $\boldsymbol{\alpha}^1$ to a point in the interior of $\Delta^n$.
Algorithm:
  • For $t = 1, \ldots, T$, repeat:
      – For all $i, y$, calculate $\nabla_{i,y} = \frac{\partial Q(\boldsymbol{\alpha}^t)}{\partial \alpha_{i,y}}$
      – For all $i, y$, update $\alpha_{i,y}^{t+1} \propto \alpha_{i,y}^t e^{-\eta\nabla_{i,y}}$
Output: Final parameters $\boldsymbol{\alpha}^{T+1}$.
```

Figure 1. A general batch EG Algorithm for minimizing $Q(\boldsymbol{\alpha})$ subject to $\boldsymbol{\alpha} \in \Delta^n$.

```
Inputs: Examples $\{(x_i, y_i)\}_{i=1}^n$, learning rate $\eta > 0$.
Initialization: Set $\boldsymbol{\alpha}^1$ to a point in the interior of $\Delta^n$.
Algorithm:
  • For $t = 1, \ldots, T$, repeat:
      – For $i = 1, \ldots, n$
          * For all $y$, calculate
            $\nabla_{i,y} = \frac{\partial}{\partial \alpha_{i,y}} Q(\boldsymbol{\alpha}_1^{t+1}, \ldots, \boldsymbol{\alpha}_{i-1}^{t+1}, \boldsymbol{\alpha}_i^t, \ldots, \boldsymbol{\alpha}_n^t)$
          * For all $y$, update $\boldsymbol{\alpha}_{i,y}^{t+1} \propto \boldsymbol{\alpha}_{i,y}^t e^{-\eta\nabla_{i,y}}$.
Output: Final parameters $\boldsymbol{\alpha}^{T+1}$.
```

Figure 2. A general online EG Algorithm for minimizing $Q(\boldsymbol{\alpha})$ subject to $\boldsymbol{\alpha} \in \Delta^n$.

Next, we consider a more general class of divergence measures: Bregman divergences. Given a convex function $\hat{Q}(\boldsymbol{\alpha})$, the Bregman divergence between $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ is defined as

$$B_{\hat{Q}}[\boldsymbol{\alpha} \| \boldsymbol{\beta}] = \hat{Q}(\boldsymbol{\alpha}) - \hat{Q}(\boldsymbol{\beta}) - \nabla\hat{Q}(\boldsymbol{\beta}) \cdot (\boldsymbol{\alpha} - \boldsymbol{\beta})$$

Convexity of $\hat{Q}$ implies $B_{\hat{Q}}[\boldsymbol{\alpha} \| \boldsymbol{\beta}] \geq 0$ for all $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \Delta^n$.

Note that the Bregman divergence with $\hat{Q}(\boldsymbol{\alpha}) = \sum_{i,y} \alpha_{i,y} \log \alpha_{i,y}$ is the KL divergence. We shall also be interested in the Mahalanobis distance

$$M_A[\boldsymbol{\alpha} \| \boldsymbol{\beta}] = \frac{1}{2}(\boldsymbol{\alpha} - \boldsymbol{\beta})^T A(\boldsymbol{\alpha} - \boldsymbol{\beta})$$

which is the Bregman divergence for $\hat{Q}(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^T A \boldsymbol{\alpha}$. We also use the L_p norm defined for $x \in \Re^m$ as $\|x\|_p = \sqrt[p]{\sum_i |x_i|^p}$. The L_∞ norm is $\|x\|_\infty = \max_i |x_i|$.

4.2. Convergence of the Batch Algorithm

We now provide a simple convergence proof for the LLEG-Batch algorithm. We begin with a useful lemma which applies to minimization of *any* convex function $\hat{Q}(\boldsymbol{\alpha})$ subject to the constraint $\boldsymbol{\alpha} \in \Delta^n$.

Lemma 1 *Consider the algorithm in Fig. 1 applied to a convex function $\hat{Q}(\boldsymbol{\alpha})$. If $\eta > 0$ is chosen such that for all t*

$$D[\boldsymbol{\alpha}^{t+1} \| \boldsymbol{\alpha}^t] \geq \eta B_{\hat{Q}}[\boldsymbol{\alpha}^{t+1} \| \boldsymbol{\alpha}^t]$$

then it follows that for all t

$$\hat{Q}(\boldsymbol{\alpha}^t) - \hat{Q}(\boldsymbol{\alpha}^{t+1}) \geq \frac{1}{\eta} D[\boldsymbol{\alpha}^t \| \boldsymbol{\alpha}^{t+1}]$$

The proof is given in App. A. Note that if we choose an η such that for all $\mathbf{p}, \mathbf{q} \in \Delta^n$ we have $D[\mathbf{p} \| \mathbf{q}] \geq \eta B_{\hat{Q}}[\mathbf{p} \| \mathbf{q}]$, then the condition in the lemma is clearly satisfied. Hence the condition can be thought of as a relation between the divergence measures D and $B_{\hat{Q}}$. For the definitions of $\hat{Q}$ used in log-linear and max-margin parameter estimation, we will show that it is relatively simple to define constraints on the values of η such that the condition holds for all $\mathbf{p}, \mathbf{q}$.

Note that $D[\mathbf{p} \| \mathbf{q}] \geq 0$ for all $\mathbf{p}, \mathbf{q} \in \Delta^n$, so the lemma implies that for an appropriately chosen η, the EG updates always decrease the objective $\hat{Q}(\boldsymbol{\alpha})$. We next show that for the log-linear dual $Q(\boldsymbol{\alpha})$, it is easy to choose an η such that the conditions of the above lemma are satisfied.

Lemma 2 *Define $|A|_\infty$ to be the maximum magnitude of any element of A.*[3] *Then for any learning rate $0 < \eta \leq \frac{1}{1+|A|_\infty}$, the LLEG-Batch updates result in a monotone improvement of the objective $Q(\boldsymbol{\alpha})$:*

$$Q(\boldsymbol{\alpha}^t) - Q(\boldsymbol{\alpha}^{t+1}) \geq \frac{1}{\eta} D[\boldsymbol{\alpha}^t \| \boldsymbol{\alpha}^{t+1}]$$

Proof: *For the $Q(\boldsymbol{\alpha})$ defined in Eq. 2, we have*

$$B_Q[\boldsymbol{\alpha}^{t+1} \| \boldsymbol{\alpha}^t] = D[\boldsymbol{\alpha}^{t+1} \| \boldsymbol{\alpha}^t] + M_A[\boldsymbol{\alpha}^{t+1} \| \boldsymbol{\alpha}^t]$$

Simple algebra yields

$$M_A[\boldsymbol{\alpha}^{t+1} \| \boldsymbol{\alpha}^t] \leq \frac{|A|_\infty}{2} \|\boldsymbol{\alpha}^{t+1} - \boldsymbol{\alpha}^t\|_1^2$$

Now use the inequality (see Cover and Thomas (1991), p. 300) $D[p_1 \| p_2] \geq \frac{1}{2} \|p_1 - p_2\|_1^2$ to obtain

$$M_A[\boldsymbol{\alpha}^{t+1} \| \boldsymbol{\alpha}^t] \leq |A|_\infty D[\boldsymbol{\alpha}^{t+1} \| \boldsymbol{\alpha}^t]$$

So for the Bregman divergence of $Q(\boldsymbol{\alpha})$ we obtain

$$B_Q[\boldsymbol{\alpha}^{t+1} \| \boldsymbol{\alpha}^t] \leq (1 + |A|_\infty) D[\boldsymbol{\alpha}^{t+1} \| \boldsymbol{\alpha}^t]$$

The condition of Lemma 1 is satisfied by choosing $0 < \eta \leq \frac{1}{1+|A|_\infty}$ and thus Lemma 2 follows. □

The above can be used to show convergence of the LLEG-Batch algorithm (see proof in App. B).

Lemma 3 *For any $0 < \eta \leq \frac{1}{1+|A|_\infty}$, the LLEG-Batch algorithm converges to $\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha} \in \Delta^n} Q(\boldsymbol{\alpha})$.*

4.3. Convergence of the Online Algorithm

In this section we prove convergence of the online algorithm, as shown in Fig. 2, when applied to the dual log-linear problem. As in Lemma 2 we can show that for this update, if we choose $0 < \eta \leq \frac{1}{1+|A|_\infty}$, the objective decreases in a monotone fashion. Define

$$Q_i^t(\boldsymbol{\alpha}_i) = Q(\boldsymbol{\alpha}_1^{t+1}, \ldots, \boldsymbol{\alpha}_{i-1}^{t+1}, \boldsymbol{\alpha}_i, \boldsymbol{\alpha}_{i+1}^t, \ldots, \boldsymbol{\alpha}_n^t)$$

The following property then holds:

$$Q_i^t(\boldsymbol{\alpha}_i^t) - Q_i^t(\boldsymbol{\alpha}_i^{t+1}) \geq \frac{1}{\eta} D[\boldsymbol{\alpha}_i^t \| \boldsymbol{\alpha}_i^{t+1}] \tag{4}$$

This follows because we can directly apply Lemma 2 to the function $Q_i^t(\boldsymbol{\alpha}_i)$. The Bregman distance corresponding to $Q_i^t(\boldsymbol{\alpha}_i)$ is of the form $B_{Q_i^t}[\boldsymbol{\alpha}_i^{t+1} \| \boldsymbol{\alpha}_i^t] = D[\boldsymbol{\alpha}_i^{t+1} \| \boldsymbol{\alpha}_i^t] + M_{A_{i,i}}[\boldsymbol{\alpha}_i^{t+1} \| \boldsymbol{\alpha}_i^t]$ where $A_{i,i}$ is a submatrix of A, and clearly $|A_{i,i}|_\infty \leq |A|_\infty$. Thus a similar proof to that in Lemma 2 can be used to show the result in Eq. 4. For any t, if we sum Eq. 4 over $i = 1 \ldots n$, we obtain

$$Q(\boldsymbol{\alpha}^t) - Q(\boldsymbol{\alpha}^{t+1}) \geq \frac{1}{\eta} \sum_{i=1}^n D[\boldsymbol{\alpha}_i^t \| \boldsymbol{\alpha}_i^{t+1}] = \frac{1}{\eta} D[\boldsymbol{\alpha}^t \| \boldsymbol{\alpha}^{t+1}]$$

We can then show convergence of the online algorithm to the optimal value $\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha}} Q(\boldsymbol{\alpha})$ using a very similar proof to that of Lemma 3.

4.4. Convergence Rate of the Batch Algorithm

In addition to proving convergence in the limit, it is relatively straightforward to give a bound on the rate of convergence for the batch algorithm. We use a similar proof technique to that of Kivinen and Warmuth (1997), in particular using the following lemma:

Lemma 4 *[Kivinen and Warmuth, 1997] For any convex function $Q(\boldsymbol{\alpha})$ over Δ^n, for any $\mathbf{u} \in \Delta^n$, if $\boldsymbol{\alpha}^{t+1}$ is derived from $\boldsymbol{\alpha}^t$ using the EG updates with a learning rate η, then*

$$\eta Q(\boldsymbol{\alpha}^t) - \eta Q(\mathbf{u}) \leq D[\mathbf{u} \| \boldsymbol{\alpha}^t] - D[\mathbf{u} \| \boldsymbol{\alpha}^{t+1}] + D[\boldsymbol{\alpha}^t \| \boldsymbol{\alpha}^{t+1}]$$

We can now use the above lemma and the bound on $D[\boldsymbol{\alpha}^t \| \boldsymbol{\alpha}^{t+1}]$ from Lemma 2 to give

$$\eta Q(\boldsymbol{\alpha}^{t+1}) - \eta Q(\mathbf{u}) \leq D[\mathbf{u} \| \boldsymbol{\alpha}^t] - D[\mathbf{u} \| \boldsymbol{\alpha}^{t+1}]$$

[3] i.e., $|A|_\infty = \max_{(i,y),(j,z)} |A_{(i,y),(j,z)}|$.

Summing this over $t = 1, \ldots, T$ gives

$$\eta \sum_{t=1}^{T} Q(\boldsymbol{\alpha}^{t+1}) - \eta T Q(\mathbf{u}) \leq D[\mathbf{u}\|\boldsymbol{\alpha}^1] - D[\mathbf{u}\|\boldsymbol{\alpha}^{T+1}]$$

Because $Q(\boldsymbol{\alpha}^t)$ is monotone decreasing, we have $TQ(\boldsymbol{\alpha}^{T+1}) \leq \sum_{t=1}^{T} Q(\boldsymbol{\alpha}^{t+1})$ and simple algebra gives

$$Q(\boldsymbol{\alpha}^{T+1}) \leq Q(\mathbf{u}) + \frac{D[\mathbf{u}\|\boldsymbol{\alpha}^1] - D[\mathbf{u}\|\boldsymbol{\alpha}^{T+1}]}{\eta T}$$

If we take $\mathbf{u}$ to be $\boldsymbol{\alpha}^*$, the optimum of D-LL, we have $Q(\boldsymbol{\alpha}^*) \leq Q(\boldsymbol{\alpha}^{T+1})$ and

$$Q(\boldsymbol{\alpha}^{T+1}) \leq Q(\boldsymbol{\alpha}^*) + \frac{D[\boldsymbol{\alpha}^*\|\boldsymbol{\alpha}^1]}{\eta T}$$

Thus to get within ϵ of the optimum, we need $O(\frac{1}{\eta\epsilon})$ iterations.

4.5. Convergence for Max-Margin Learning

Bartlett et al. (2004) considered a batch EG algorithm for max-margin parameter estimation, which we will call the MMEG algorithm. The result in Lemma 1 applies to any convex function $\hat{Q}(\boldsymbol{\alpha})$; this section gives a convergence proof for the MMEG algorithm.

The MMEG algorithm involves optimization of a dual problem with the same structure as D-LL, only with $Q(\boldsymbol{\alpha})$ replaced by $Q_{MM}(\boldsymbol{\alpha})$, where

$$Q_{MM}(\boldsymbol{\alpha}) = \mathbf{b}^T\boldsymbol{\alpha} + \frac{1}{2}\boldsymbol{\alpha}^T A \boldsymbol{\alpha}$$

The matrix A is defined as in Sec. 2 and $\mathbf{b}$ is a vector.[4] The MMEG algorithm is then the batch EG algorithm in Fig. 1 applied to the function $Q_{MM}(\boldsymbol{\alpha})$. The difference from D-LL is that the entropy term is replaced by a linear term. Since the Bregman divergence corresponding to the linear term is identically zero, we have[5] that Lemma 1 is satisfied for any η such that $0 < \eta \leq \frac{1}{|A|_\infty}$. We can then use Lemma 4 to give a rate of convergence for the MMEG algorithm. In particular, for $\eta = \frac{1}{|A|_\infty}$, after T iterations of the MMEG algorithm we have

$$Q_{MM}(\mathbf{u}) \leq Q_{MM}(\boldsymbol{\alpha}^{T+1}) \leq Q_{MM}(\mathbf{u}) + \gamma T^{-1} D[\mathbf{u}\|\boldsymbol{\alpha}^1]$$

where $\mathbf{u} = \arg\min_{\boldsymbol{\alpha}\in\Delta^n} Q_{MM}(\boldsymbol{\alpha})$, and $\gamma = |A|_\infty$. The convergence result in Bartlett et al. (2004) has a value of $\gamma = O(B + \lambda_A)$ where $B \approx n|A|_\infty$ and λ_A is the largest eigenvalue of A; our proof has removed the constant B, which is $O(n)$ where n is the number of training examples. We have also replaced λ_A with $|A|_\infty$.

5. Structured Prediction with LLEG

We now describe how the EG updates can be applied to structured prediction problems, for example parameter estimation in CRFs or natural language parsing. In structured problems the label set $\mathcal{Y}$ is typically very large, but labels can have useful internal structure. As one example, in CRFs each label y is an m-dimensional vector specifying the labeling of all m vertices in a graph. In parsing each label y is an entire parse tree.

We follow the framework for structured problems described by Bartlett et al. (2004). Each label y is defined to be a set of *parts*. We use R to refer to the set of all possible parts. We make the assumption that the feature vector for an entire label y decomposes into a sum over feature vectors for individual parts as follows: $\boldsymbol{\phi}(x, y) = \sum_{r\in y} \boldsymbol{\phi}(x, r)$. Note that we have overloaded $\boldsymbol{\phi}$ to apply to either labels y or parts r.

The label set $\mathcal{Y}$ can be extremely large in structured prediction problems, precluding a direct implementation of the LLEG-Batch and LLEG-Online algorithms. However, in this section we describe an approach that does allow an efficient implementation of the algorithms in several cases; the approach is similar to that described in Bartlett et al. (2004). Instead of manipulating the dual variables $\boldsymbol{\alpha}_i^t$ for each t, i directly, we will make use of alternative data structures $\boldsymbol{\theta}_i^t$ for all t, i. Each $\boldsymbol{\theta}_i^t$ is a vector of real values $\theta_{i,r}^t$ for all $r \in R$. The $\boldsymbol{\theta}_i^t$ variables implicitly define regular dual values $\boldsymbol{\alpha}_i^t = \boldsymbol{\alpha}(\boldsymbol{\theta}_i^t)$ where the mapping between $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$ is defined as $\alpha_{i,y}(\boldsymbol{\theta}_i^t) \propto \exp\{\sum_{r\in y} \theta_{i,r}^t\}$. To see how the $\boldsymbol{\theta}_i^t$ variables can be updated, note that the EG updates for the log-linear dual involve the gradient expression

$$\nabla_{i,y}^t = 1 + \log \alpha_{i,y}^t + \frac{1}{C}\mathbf{w}(\boldsymbol{\alpha}^t)\cdot(\boldsymbol{\phi}(x_i, y_i) - \boldsymbol{\phi}(x_i, y))$$

Equivalently, we can perform the EG updates using

$$\nabla_{i,y}^t = \log \alpha_{i,y}^t - \frac{1}{C}\sum_{r\in y} \mathbf{w}(\boldsymbol{\alpha}^t)\cdot\boldsymbol{\phi}(x_i, r)$$

because the expression $\mathbf{w}(\boldsymbol{\alpha}^t)\cdot\boldsymbol{\phi}(x_i, y_i)$ is constant w.r.t. y, and therefore cancels in the EG updates. Consider the following update to the $\boldsymbol{\theta}$ variables:

$$\theta_{i,r}^{t+1} = \theta_{i,r}^t - \eta\left(\theta_{i,r}^t - \frac{1}{C}\mathbf{w}(\boldsymbol{\alpha}(\boldsymbol{\theta}^t))\cdot\boldsymbol{\phi}(x, r)\right)$$

The following property of these updates can then

[4] More specifically, $b_{i,y}$ is defined to be a measure of the *loss* for label y on the i'th example.

[5] The proof is similar to the proof of Lemma 2, but with $B_Q[\boldsymbol{\alpha}^{t+1}\|\boldsymbol{\alpha}^t] = D[\boldsymbol{\alpha}^{t+1}\|\boldsymbol{\alpha}^t] + M_A[\boldsymbol{\alpha}^{t+1}\|\boldsymbol{\alpha}^t]$ replaced by $B_{Q_{MM}}[\boldsymbol{\alpha}^{t+1}\|\boldsymbol{\alpha}^t] = M_A[\boldsymbol{\alpha}^{t+1}\|\boldsymbol{\alpha}^t]$.

be shown. Given that $\boldsymbol{\alpha}_i^t = \boldsymbol{\alpha}(\boldsymbol{\theta}_i^t)$, it follows that $\boldsymbol{\alpha}_i^{t+1} = \boldsymbol{\alpha}(\boldsymbol{\theta}_i^{t+1})$, where $\boldsymbol{\alpha}_i^{t+1}$ is derived from $\boldsymbol{\alpha}_i^t$ using the regular EG updates, and $\boldsymbol{\theta}_i^{t+1}$ is derived from $\boldsymbol{\theta}_i^t$ using the above updates. Thus our LLEG algorithms can be re-stated in terms of the $\boldsymbol{\theta}$ variables, using these updates. The main computational challenge is computing the parameter vector $\mathbf{w}(\boldsymbol{\alpha}(\boldsymbol{\theta}^t))$. This can be achieved if the problem is such that *marginals* can be computed efficiently from the $\boldsymbol{\theta}_i$ parameters; for example, in CRFs belief propagation can be used to compute the required marginals. See Bartlett et al. (2004) for the details of how $\mathbf{w}(\boldsymbol{\alpha}(\boldsymbol{\theta}^t))$ can be computed assuming that marginals can be computed efficiently.

6. Related Work

As mentioned earlier, several works have addressed optimizing log-linear models via the convex dual of the likelihood-maximization problem. Earlier works (Jaakkola & Haussler, 1999; Keerthi et al., 2005; Zhu & Hastie, 2001) treated the logistic regression model, a simpler version of a CRF. In the binary logistic regression case, there is essentially one parameter α_i with constraint $0 \leq \alpha_i \leq 1$ and therefore simple line search methods can be used for optimization. Minka (2003) shows empirical results which show this approach performs similarly to conjugate gradient. Recent work (Shalev-Shwartz & Singer, 2006) presents a more general study of dual algorithms in the online setting.

The problem becomes much harder when $\boldsymbol{\alpha}_i$ is constrained to be a distribution over many labels, as in the case discussed here. Recently, Memisevic (2006) addressed this setting, and suggests optimizing $\boldsymbol{\alpha}_i$ by transferring probability mass between two labels y_1, y_2 while keeping the distribution normalized.

Convergence results for EG on batch problems have been given in the optimization literature. Beck and Teboulle (2003) describe convergence results which apply to quite general definitions of $Q(\boldsymbol{\alpha})$, but which have $1/\sqrt{T}$ convergence rates (slower than our results of $1/T$). Also, their work considers optimization over a single simplex, and does not consider online-like algorithms such as the one we have presented.

7. Experiments

We compared our LLEG-Online algorithm to Conjugate Gradient (CG) and L-BFGS algorithms for two classification tasks:[6] a logistic regression model for multiclass learning, and a log-linear model for a structured natural language dependency parsing task.

Although EG is guaranteed to converge for an appropriately chosen η, it turns out to be beneficial to use an adaptive learning rate. Here we use a crude strategy: for every sample point we start out with some η_0 and halve it until the objective is decreased. More refined line-search methods are likely to improve performance.

We measure the performance of each training algorithm as a function of the amount of computation spent. Specifically, we measure computation in terms of the number of times each training example is visited. For CG/L-BFGS, a cost of n is incurred every time the gradient or objective function is evaluated; note that because CG/L-BFGS use a line search, each iteration may involve several such evaluations. For EG, the cost of an example is the number of objective evaluations on that example. This corresponds to the number of different η values tested on this example.

CG/L-BFGS and EG optimize the primal and dual problems, respectively, and by convex duality are guaranteed to converge to the same value. To compare EG and CG/L-BFGS, we can use the EG weight vector $\mathbf{w}(\boldsymbol{\alpha}^t)$ to compute a primal value. Note that EG does not explicitly minimize the primal objective function, so the EG primal will not necessarily decrease at every iteration. Nevertheless, our experiments show that the EG primal decreases much faster in early iterations than the CG/L-BFGS primal.

7.1. Multiclass classification

We conducted multiclass classification experiments on a subset of the MNIST classification task. Examples in this dataset are images of handwritten digits represented as 784-dimensional vectors. We used a set of 10k examples, split into 7k for training, 1.5k for development, and 1.5k for test. We define a ten-class logistic regression model where $p(y|x) \propto e^{\mathbf{x}\cdot\mathbf{w}_y}$ and $\mathbf{x}, \mathbf{w}_y \in \Re^{784}, y \in \{1, \ldots, 10\}$.[7]

Figure 3 shows the primal objective for EG and CG/L-BFGS, and the dual objective for EG. As expected, the primal and dual converge to the same value. Furthermore, the primal value for EG converges considerably faster than the CG/L-BFGS one. Also shown is classification accuracy for both algorithms. Again, it can be seen that EG converges considerably faster.

[6] For L-BFGS, we used C. Zhu, R.H. Byrd, P. Lu, and J. Nocedal's code (www.ece.northwestern.edu/∼nocedal/) and L. Stewart's wrapper (www.cs.toronto.edu/∼liam/). For CG, Section 7.1 used J. Rennie's code (people.csail.mit.edu/jrennie/matlab/) and Section 7.2 used W. Hager and Z. Zhang's package (www.math.ufl.edu/∼hager/papers/CG/).

[7] The regularization parameter C was chosen by optimizing on the development set for all experiments.

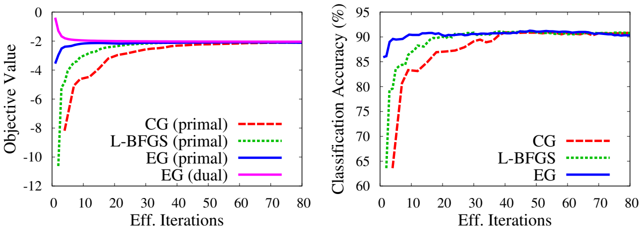

Figure 3. Results on the MNIST learning task. The left panel shows the primal objective for both EG and CG/L-BFGS, and the dual objective for EG. The EG,CG/L-BFGS primals are increasing functions and the EG dual is the decreasing function. The right panel shows the validation error for EG and CG/L-BFGS. The X axis counts the number of effective iterations over the entire data set.

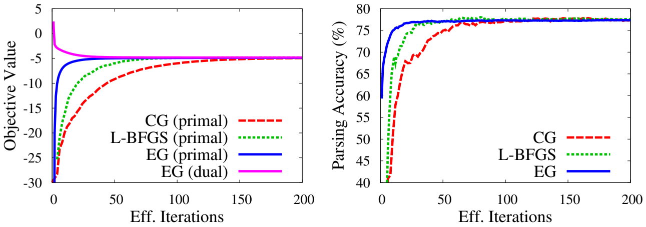

Figure 4. Results on the dependency parsing task. The figures correspond to those in Fig. 3.

7.2. Structured learning - Dependency Parsing

Parsing of natural language sentences is a challenging structured learning task. Dependency parsing (McDonald et al., 2005) is a simplified form of parsing where the goal is to map sentences x into projective directed spanning trees over the set of words $\mathcal{Y}(x)$. Assuming we have a function $\boldsymbol{\phi}(x, h, m)$ which assigns a feature vector to pairs of words (h, m), we can use a weight vector $\mathbf{w}$ to score a given tree y by $\mathbf{w} \cdot \sum_{(h,m)\in y} \boldsymbol{\phi}(x, h, m)$. Thus we are in the setting of the log-linear model addressed in the current paper. Dependency parsing corresponds to a structured problem where the parts r are dependencies (h, m); the approach described in Sec. 5 can be applied efficiently to dependency structures.[8]

In the experiment below we use a feature set $\boldsymbol{\phi}(x, h, m)$ similar to that in (McDonald et al., 2005). We report result on the Slovene dataset which is part of the CoNLL-X Shared Task on multilingual dependency parsing (Buchholz & Marsi, 2006). The data consists of $1,234$ sentences with $22,949$ tokens. We leave out 200 sentences and report accuracy (rate of correct edges in a predicted parse tree) on those. Fig. 4 reports the performance of CG/L-BFGS and EG on this task. It can be seen that EG converges faster than CG/L-BFGS both in terms of objective and error measures.

8. Conclusion

We presented a novel algorithm for large scale learning of log-linear models, which provably converges to the optimum. Most of our proofs rely on a relation between B_Q and the KL divergence. This relation holds for max-margin learning as well, a fact which simplifies previous results in this setting. We expect a similar analysis to hold for other functions Q. Finally, the current work does not provide convergence rates for the online algorithm. It remains an interesting challenge to obtain a $1/T$ convergence rate for this case.

Acknowledgments

X.C. is supported by the Catalan Ministry of Innovation, Universities and Enterprise. A.G. is supported by a fellowship from the Rothschild Foundation - Yad Hanadiv. T.K. was funded by a grant from the NSF (DMS-0434222) and a grant from NTT, Agmt. Dtd. 6/21/1998. M.C. was funded by NSF grants 0347631 and DMS-0434222.

References

Baker, J. (1979). Trainable grammars for speech recognition. *97th meeting of the Acoustical Society of America.*

Bartlett, P., Collins, M., Taskar, B., & McAllester, D. (2004). Exponentiated gradient algorithms for large–margin structured classification. *NIPS.*

Beck, A., & Teboulle, M. (2003). Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, *31*, 167–175.

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

Collins, M., Schapire, R., & Singer, Y. (2002). Logistic regression, adaboost and bregman distances. *Machine Learning*, *48*, 253–285.

Cover, T., & Thomas, J. (1991). *Elements of information theory.* Wiley.

Jaakkola, T., & Haussler, D. (1999). Probabilistic kernel regression models. *Proc. of AISTATS.*

Keerthi, S., Duan, K., Shevade, S., & Poo, A. N. (2005). A fast dual algorithm for kernel logistic regression. *Machine Learning*, *61*, 151–165.

Kivinen, J., & Warmuth, M. (1997). Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, *132*, 1–63.

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

[8] The required marginals can be computed efficiently using a variant of the inside-outside algorithm (Baker, 1979).

Lebanon, G., & Lafferty, J. (2001). Boosting and maximum likelihood for exponential models. *NIPS*.

McDonald, R., Crammer, K., & Pereira, F. (2005). Online large-margin training of dependency parsers. *Proc. of the 43rd Annual Meeting of the ACL*.

Memisevic, R. (2006). *Dual optimization of conditional probability models* (Technical Report). Univ. of Toronto.

Minka, T. (2003). *A comparison of numerical optimizers for logistic regression* (Technical Report). CMU.

Platt, J. (1998). Fast training of support vector machines using sequential minimal optimization. In *Advances in kernel methods - support vector learning*. MIT Press.

Sha, F., & Pereira, F. (2003). Shallow parsing with conditional random fields. *Proc. of HLT–NAACL*.

Shalev-Shwartz, S., & Singer, Y. (2006). Convex repeated games and fenchel duality. *NIPS*.

Taskar, B., Guestrin, C., & Koller, D. (2003). Max margin markov networks. *NIPS*.

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

Zhu, J., & Hastie, T. (2001). Kernel logistic regression and the import vector machine. *NIPS*.

A. Proof of Lemma 1

Given an $\boldsymbol{\alpha}^t$, the EG update is $\alpha^{t+1}_{i,y} = \frac{1}{Z_i^t}\alpha^t_{i,y}e^{-\eta\nabla^t_{i,y}}$, where $\nabla^t_{i,y} = \frac{\partial Q(\boldsymbol{\alpha}^t)}{\partial \alpha_{i,y}}$ and $Z_i^t = \sum_{\hat{y}}\alpha^t_{i,\hat{y}}e^{-\eta\nabla^t_{i,y}}$. Simple algebra yields

$$\sum_i \big(D[\boldsymbol{\alpha}_i^t\|\boldsymbol{\alpha}_i^{t+1}]+D[\boldsymbol{\alpha}_i^{t+1}\|\boldsymbol{\alpha}_i^t]\big) = \eta\sum_{i,y}(\alpha^t_{i,y}-\alpha^{t+1}_{i,y})\nabla^t_{i,y}$$

Equivalently, using the notation for KL divergence between multiple distributions:

$$D[\boldsymbol{\alpha}^t\|\boldsymbol{\alpha}^{t+1}]+D[\boldsymbol{\alpha}^{t+1}\|\boldsymbol{\alpha}^t] = \eta(\boldsymbol{\alpha}^t-\boldsymbol{\alpha}^{t+1})\cdot\nabla Q(\boldsymbol{\alpha}^t)$$

The definition of Bregman divergence B_Q implies

$$-\eta B_Q[\boldsymbol{\alpha}^{t+1}\|\boldsymbol{\alpha}^t]+D[\boldsymbol{\alpha}^t\|\boldsymbol{\alpha}^{t+1}]+D[\boldsymbol{\alpha}^{t+1}\|\boldsymbol{\alpha}^t] = \eta(Q(\boldsymbol{\alpha}^t)-Q(\boldsymbol{\alpha}^{t+1}))$$

It can then be seen that $D[\boldsymbol{\alpha}^{t+1}\|\boldsymbol{\alpha}^t] \geq \eta B_Q[\boldsymbol{\alpha}^{t+1}\|\boldsymbol{\alpha}^t]$ implies $\eta(Q(\boldsymbol{\alpha}^t)-Q(\boldsymbol{\alpha}^{t+1})) \geq D[\boldsymbol{\alpha}^t\|\boldsymbol{\alpha}^{t+1}]$.

B. Proof of Lemma 3

The proof is similar to that of Lemma 2 in Collins et al. (2002). The choice of η implies that Lemma 2 in the current paper holds. The sequence $Q(\boldsymbol{\alpha}^t)$ is then monotone and bounded from below, and therefore convergent. Its difference series thus converges to zero. Together with Lemma 2 this implies: $\lim_{t\to\infty} D[\boldsymbol{\alpha}^t\|\boldsymbol{\alpha}^{t+1}] = 0$. Denote the mapping from $\boldsymbol{\alpha}^t$ to $\boldsymbol{\alpha}^{t+1}$ by $f(\boldsymbol{\alpha}^t) = \boldsymbol{\alpha}^{t+1}$. If we define $F(\boldsymbol{\alpha}^t) = D[\boldsymbol{\alpha}^t\|f(\boldsymbol{\alpha}^t)]$ then $\lim_{t\to\infty} F(\boldsymbol{\alpha}^t) = 0$. The $\boldsymbol{\alpha}^t$ values lie in a compact space, and therefore must have a subsequence that converges to some point $\boldsymbol{\alpha}^\infty \in \Delta^n$. Continuity of F implies that $F(\boldsymbol{\alpha}^\infty) = 0$. Because the KL divergence is zero iff its arguments are identical, this implies that $\boldsymbol{\alpha}^\infty$ is a fixed point of the EG updates, i.e., $\boldsymbol{\alpha}^\infty = f(\boldsymbol{\alpha}^\infty)$.

We next show that $\boldsymbol{\alpha}^\infty$ is necessarily the dual optimum $\boldsymbol{\alpha}^*$. By the KKT conditions, if a pair $(\boldsymbol{\alpha}^*, \lambda^*)$ is such that $\boldsymbol{\alpha}^*$ is in the interior of Δ^n and $\frac{\partial Q(\boldsymbol{\alpha}^*)}{\partial\alpha_{i,y}} = \lambda_i^*$, then $Q(\boldsymbol{\alpha}^*)$ is the optimal value of the dual. To show that $\boldsymbol{\alpha}^\infty = \boldsymbol{\alpha}^*$, we need the following lemma, which states that $\boldsymbol{\alpha}^\infty$ is in the strict interior of Δ^n. The lemma is proved in App. C.

Lemma 5 *There are constants c_1, c_2 such that for all t, i and y we have $0 < c_1 \leq \alpha^t_{i,y} \leq c_2 < 1$. Thus, the limit point $\boldsymbol{\alpha}^\infty$ satisfies $0 < \alpha^\infty_{i,y} < 1$ for all i, y.*

The above lemma implies that all components of $\boldsymbol{\alpha}^\infty$ are non-zero. Since $f(\boldsymbol{\alpha}^\infty) = \boldsymbol{\alpha}^\infty$ it follows that

$$\frac{\partial Q(\boldsymbol{\alpha}^\infty)}{\partial\alpha_{i,y}} = -\frac{1}{\eta}\log\sum_{\hat{y}}\alpha^\infty_{i,\hat{y}}e^{-\eta\frac{\partial Q(\boldsymbol{\alpha}^\infty)}{\partial\alpha_{i,\hat{y}}}} \equiv c_i$$

We can now obtain a point that satisfies the KKT conditions by defining $\boldsymbol{\alpha}^* = \boldsymbol{\alpha}^\infty$ and $\lambda_i^* = c_i$. The pair $(\boldsymbol{\alpha}^*, \lambda^*)$ clearly satisfies the KKT conditions above and the condition $\boldsymbol{\alpha}^* \in \Delta^n$. It is thus the optimum of D-LL in Eq. 3. The strict convexity of $Q(\boldsymbol{\alpha})$ implies that the optimal point is unique. Using this we can show that the entire $\boldsymbol{\alpha}^t$ sequence also converges to $\boldsymbol{\alpha}^*$ using an argument similar to that in Collins et al. (2002).

C. Proof of Lemma 5

For the D-LL problem, the update can be written as $\alpha^{t+1}_{i,y} \propto e^{(1-\eta)\log\alpha^t_{i,y}-\frac{\eta}{C}\mathbf{w}(\boldsymbol{\alpha}^t)\cdot\boldsymbol{\psi}_{i,y}}$. Starting at $\boldsymbol{\alpha}^1$ and reapplying this update, we can express $\log\alpha^{t+1}_{i,y}$ as

$$(1-\eta)^t\log\alpha^1_{i,y}+\frac{1}{C}\sum_{k=1}^{t}\eta(1-\eta)^k\mathbf{w}(\boldsymbol{\alpha}^k)\cdot\boldsymbol{\psi}_{i,y}+c_{t,i}$$

where $c_{t,i}$ is not dependent on y. From the definition of $\mathbf{w}(\boldsymbol{\alpha})$ in Eq. 1 we can see that $\frac{1}{C}|\mathbf{w}(\boldsymbol{\alpha}^k)\cdot\boldsymbol{\psi}_{i,y}| \leq a$ for some positive constant a. For $\eta \leq 1$ we have $\sum_{k=1}^{t}\eta(1-\eta)^k \leq 1$. Thus, for all t, i, y, z, we have $|\log\frac{\alpha^t_{i,y}}{\alpha^t_{i,z}}| \leq c$ for some constant c. Since $\boldsymbol{\alpha}^t_i$ must be normalized it follows that for all t, i, y we have $0 < \frac{1}{1+Ke^c} \leq \alpha^t_{i,y} \leq \frac{1}{1+Ke^{-c}} < 1$ where $K = |\mathcal{Y}|-1$.