An Online Learning Algorithm for Bilinear Models

Yuanbin Wu YBWU@CS.ECNU.EDU.CN
Shiliang Sun SLSUN@CS.ECNU.EDU.CN
Shanghai Key Laboratory of Multidimensional Information Processing
Department of Computer Science and Technology, East China Normal University

Abstract

We investigate the bilinear model, which is a matrix form linear model with the rank 1 constraint. A new online learning algorithm is proposed to train the model parameters. Our algorithm runs in the manner of online mirror descent, and gradients are computed by the power iteration. To analyze it, we give a new second order approximation of the squared spectral norm, which helps us to get a regret bound. Experiments on two sequential labelling tasks give positive results.

1. Introduction

In supervised classification, linear models are important and fundamental. Features are packed into a vector, and a weight in the same vector space is used to vote the importance of different features. However, in some applications of computer vision (Pirsiavash et al., 2009), natural language processing (Lei et al., 2014) and recommender systems (Rendle, 2010), matrices are more natural and informative than vectors to express features They can help to explore latent structures of the input space (e.g., semantic relations among features), which can potentially improve the classification performance. In this work, as a specific case, we will study the bilinear model, which is a matrix form linear model with the rank 1 constraint.

The rank constraint brings difficulties both on designing and analyzing learning algorithms. We introduce a simple and fast online algorithm for the bilinear model which tries to overcome those difficulties. First, models with low rank constraints usually need singular value decomposition (SVD). The full SVD is computationally unaffordable for large scale matrices. In the case of our bilinear model, we will rely on the power iteration to compute the leading singular vectors. By the carefully selected initial value and normalization factor, we get an efficient update of singular vectors. Second, since the rank constraint is non-convex, the framework of online convex optimization (Shalev-Shwartz, 2012) is not directly applicable for analyzing the learning problem. We give a second order approximation of the squared spectral norm (Proposition 3). It serves as a complement of the strong smoothness result on the squared Schatten norm (Ball et al., 1994; Duchi et al., 2010; Kakade et al., 2012). Equipped with this result, we derive a regret bound of the algorithm.

We conduct experiments on two sequential labelling tasks: word segmentation and text chunking. The results show that the prior knowledge expressed by the matrix form feature and the new online learning algorithm can help to build efficient and competitive models.

2. Related Work

Bilinear models have been applied in computer vision (Tenenbaum & Freeman, 2000; Pirsiavash et al., 2009). Major motivations of these works are that it is more natural to represent images by matrices, and the bilinear formulation can help to reduce the number of parameters and the risk of overfitting. In natural language processing, although the matrix feature representation is not as obvious as the intensity matrix of an image, the bilinear models could also have clear physical interpretations. For example, a tensor model has been recently proposed by Lei et al. (2014) for dependency parsing. Different from their work, we give a solid formulation and analysis of the learning problem.

There are many works on low rank approximations in collaborative filtering (Srebro et al., 2005; Rennie & Srebro, 2005; Wang et al., 2013). It is popular to use the trace norm as a convex surrogate of rank constraints. In this work, we deal with a special case of hard rank constraints ($\mathrm{rank} = 1$). The analysis of our learning algorithm will show the relation to the trace norm regularization. Shalev-Shwartz et al. (2011) considered the general low rank constrained optimization problem with convex objectives. Compared with that work, we investigate the

Proceedings of the 32^{nd} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37.

dual problem which might be more efficient in the case of $\mathrm{rank} = 1$: instead of computing the leading singular vectors of a "big" gradient matrix, we incrementally compute singular vectors for a sequence of matrices, and each step only involves sparse matrix operations.

Another closely related topic is online mirror descent (Duchi et al., 2010; Kakade et al., 2012; Shalev-Shwartz, 2012). By using different strongly convex functions, it unifies many existing online learning algorithms. In fact, our algorithm (Eq. 8) runs in the manner of mirror descent. However, due to the non-convexity of rank constraints, tools from the online mirror descent framework are not readily applicable to analyzing our algorithm. To proceed, we turn to view the proposed method as a dual coordinate ascent approach (Shalev-Shwartz & Singer, 2006; Shalev-Shwartz & Kakade, 2008; Shalev-Shwartz & Zhang, 2013). It increases the dual objective incrementally, and the loss could be bounded by the weak duality.

3. The Model

3.1. Notations

For a matrix $A \in \mathsf{R}^{m\times n}$, let $\sigma(A) = [\sigma_1(A), \ldots, \sigma_l(A)]^\intercal$ be A's singular values, where $\sigma_1(A) \geq \cdots \geq \sigma_l(A)$, $l = \min\{m, n\}$. Denote $\|A\|_{\mathrm{F}}$ as the Frobenius norm, $\|A\|_2 = \sigma_1$ as the spectral norm, $\|A\|_{\mathbf{s}(p)} = \|\sigma(A)\|_p$ as the Schatten p-norm, and $\|A\|_{\mathbf{k}(k)} = \sum_{i=1}^{k} \sigma_i$ as the Ky Fan k-norm. The inner product $\langle A, B\rangle = \mathrm{Tr}(A^\intercal B)$. $A \otimes B$ is the Kronecher product. Let F be a real-valued function, and its Fenchel conjugate is denoted by F^*.

3.2. The Bilinear Model

We consider the matrix form linear classifier $h : X \mapsto Y$:

$$h(x) = \arg\max_{y\in Y} \mathrm{Tr}(W^\intercal \Phi(x, y)),$$

where X is the input space, Y is the class label set, $\Phi : X \times Y \mapsto \mathsf{R}^{m\times n}$ is the matrix-valued feature function, and $W \in \mathsf{R}^{m\times n}$ is the model parameter. When $n = 1$, we get the vector form linear model.

In practice, instead of using free W, we may be interested in models with additional constraints. On the one hand, in some applications, we have prior knowledge about semantic relations among features, which can help to improve the classification performances. We would like to encode such information both in $\Phi(x, y)$ and W. On the other hand, by imposing different matrix constraints on W, we can tailor parameters to meet the structure of the input space, which may result in more efficient and compact models.

In this paper, we will explore a specific constraint on W: the rank 1 constraint, and a bilinear model:

$$h(x) = \arg\max_{y\in Y} \alpha^\intercal \Phi(x, y)\beta, \tag{1}$$

where $\alpha \in \mathsf{R}^m, \beta \in \mathsf{R}^n$. The model parameter $W = \alpha\beta^\intercal$ is a rank 1 matrix.

The following section contains a concrete example of the bilinear formulation for sequential labelling, which is a baseline of many natural language processing tasks. We show that the rank 1 constraint appears naturally by prior knowledge, and helps to reduce the number of parameters.

3.3. An Example

For an input sentence x, the sequential labelling task outputs a label sequence $y = y_1 y_2 \ldots y_{|y|} \in Y$, where Y contains all possible such sequences. Let S be the label set, where $y_i \in S$. For simplicity, assume $S = \{\mathrm{B}, \mathrm{I}, \mathrm{O}\}$ [1].

We first review the vector form linear model:

$$h(x) = \arg\max_{y\in Y} w^\intercal \hat{\Phi}(x, y). \tag{2}$$

With the first order Markov assumption, let $\hat{\Phi}(x, y) = \sum_{i=1}^{|y|} \hat{\Phi}(x, y_i, y_{i-1})$, and $h(x)$ is computed by the standard Viterbi algorithm.

In natural language processing, $\hat{\Phi}(x, y_i, y_{i-1})$ are usually sparse vectors in a high dimensional vector space. They could be instantiated by a set of feature templates. For example, a template could be "whether ith word of x is v". It will expand to a vector, which is indexed by all possible assignments of (y_i, y_{i-1}) [2]. Here, if position i of x is indeed v and $y_i = \mathrm{B}, y_{i-1} = \mathrm{O}$, the template will expand to

$$\begin{array}{cccccccccc} & \text{BB} & \text{BI} & \textbf{BO} & \text{IB} & \text{II} & \text{IO} & \text{OB} & \text{OI} & \text{OO} \\ \big[& 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \ \big]. \end{array} \tag{3}$$

Formally, a feature template is a function $\hat{\varphi} : X \times S \times S \mapsto \{0, 1\}^{|S|^2}$,

$$\hat{\varphi}^{u,v} = P(x) \cdot \mathbb{I}\left(y_i = u, y_{i-1} = v\right), \tag{4}$$

where (u, v) is an index of vector $\hat{\varphi}(x, y_i, y_{i-1})$, $P(x)$ is a boolean function, and $\mathbb{I}$ is the indicator function. Given K templates, $\hat{\Phi}(x, y_i, y_{i-1})$ is a blocked feature vector:

$$\hat{\Phi}(x, y_i, y_{i-1}) = [\hat{\varphi}_1(x, y_i, y_{i-1})^\intercal, \ldots, \hat{\varphi}_K(x, y_i, y_{i-1})^\intercal]^\intercal.$$

Next, we show how to use a blocked diagonal matrix to represent features, and then exploit a rank 1 constraint to

[1] "B": begin, "I": inside, "O": outside.

[2] There are $|V|$ templates, where V is the vocabulary. Hence, for each x, we have a sparse (only $|x| \approx 10^1$ active entries) high dimension ($|V||S|^2 \approx 10^4$) feature vector. For bigram features (whether $\mathsf{v}_1, \mathsf{v}_2$ appears), the dimension will be in order of 10^7.

get a bilinear model. A simple observation on the feature template (4) is that $\hat{\varphi}^{u,v}$ can be decomposed:

$$\hat{\varphi}^{u,v} = P(x)\mathbb{I}(y_i = u) \cdot P(x)\mathbb{I}(y_{i-1} = v). \quad (5)$$

Accordingly, we have two separated "views" on a single feature: one from the current label, the other from the previous label. It implies that the corresponding weight could also be decomposed. Furthermore, for a template $\hat{\varphi}(x, y_i, y_{i-1})$, different instantiations of (y_i, y_{i-1}) can share weights if their "views" overlap, so the total number of parameters is reduced. The following are the details.

Define the matrix feature template $\varphi(x, y_i, y_{i-1}) = \zeta_1(x, y_i) \otimes \zeta_2(x, y_{i-1})$, where $\zeta_1(x, \cdot), \zeta_2(x, \cdot) : X \times S \mapsto \{0, 1\}^{|S|}$ have elements

$$\zeta_1^u(x, y_i) = P(x) \cdot \mathbb{I}(y_i = u)$$
$$\zeta_2^v(x, y_{i-1}) = P(x) \cdot \mathbb{I}(y_{i-1} = v).$$

The new feature template will expand to a matrix, rather than a vector. For example, now (3) is

$$\begin{array}{c} \\ \text{B} \\ \text{I} \\ \text{O} \end{array}\!\!\begin{array}{c} \begin{array}{ccc} \text{B} & \text{I} & \text{O} \end{array} \\ \begin{bmatrix} 0 & 0 & \mathbf{1} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{array} = \begin{array}{c} \\ \text{B} \\ \text{I} \\ \text{O} \end{array}\!\!\begin{array}{c} \begin{array}{c} \\ \end{array} \\ \begin{bmatrix} \mathbf{1} \\ 0 \\ 0 \end{bmatrix} \end{array} \quad \begin{array}{c} \begin{array}{ccc} \text{B} & \text{I} & \text{O} \end{array} \\ \begin{bmatrix} 0 & 0 & \mathbf{1} \end{bmatrix} \end{array}.$$

$$\varphi(x, y_i, y_{i-1}) \quad \zeta_1(x, y_i) \quad \zeta_2^\intercal(x, y_{i-1})$$

Define the matrix-valued feature function as

$$\Phi(x, y_i, y_{i-1}) \triangleq \text{diag}(\varphi_1(x, y_i, y_{i-1}), \ldots, \varphi_K(x, y_i, y_{i-1})).$$

Denoting $\Phi(x, y) = \sum_{i=1}^n \Phi(x, y_i, y_{i-1})$, we have the matrix form linear model

$$h(x) = \arg\max_{y \in Y} \text{Tr}(W^\intercal \Phi(x, y)). \quad (6)$$

Until now, we haven't changed the linear model. Indeed, (6) is equal to (2) if W is unconstrained. But it is interesting to investigate W with additional structures. For example, driven by the feature template decomposition (5), it is natural to question whether we can also decompose the weight matrix W. In other words, whether it is possible to assign weights on $\zeta_1(x, \cdot)$, $\zeta_2(x, \cdot)$, rather than $\varphi(x, \cdot, \cdot)$. These questions lead to a W with $\text{rank} = 1$ constraint, and we get a bilinear model as (1).

To clarify the decomposition of W, let's expand the two discriminant functions (1) and (2) on a single template:

$$\sum_{(u,v) \in S \times S} \alpha_u \cdot \beta_v \cdot P(x) \cdot \mathbb{I}(y_i = u) \cdot \mathbb{I}(y_{i-1} = v),$$
$$\sum_{(u,v) \in S \times S} w_{u,v} P(x) \cdot \mathbb{I}(y_i = u, y_{i-1} = v).$$

Thus, $w_{u,v} = \alpha_u \cdot \beta_v$. Since $\alpha, \beta \in \mathsf{R}^{|S|K}$, the number of parameters is $2|S|K$, which is less than $|S|^2 K$ in the case of the original linear model (assume $|S| > 2$).

Algorithm 1 Blockwise Coordinate Descent

1: $\alpha^{(0)} = \frac{\mathbf{1}}{\|\mathbf{1}\|}$; $\beta^{(0)} = \frac{\mathbf{1}}{\|\mathbf{1}\|}$; R: number of iterations
2: **for** $r = 0$ to R **do**
3: $\quad \alpha^{(r+1)} = \text{SVMsolver}(\alpha^{(r)}, \beta^{(r)})$
4: $\quad \beta^{(r+1)} = \text{SVMsolver}(\beta^{(r)}, \alpha^{(r+1)})$
5: **end for**
6: **return** $\alpha^{(R)}, \beta^{(R)}$

4. Online Learning of the Bilinear Model

4.1. The Algorithm

Let $\{(x^j, y^j)\}_{j=1}^N$ be a training set. Consider an SVM with the bilinear formulation,

$$\min_{W = \alpha\beta^\intercal \in \Omega_1} \quad \frac{1}{2}\|W\|_{\text{F}}^2 + C \sum_{j=1}^N L(W; x^j, y^j), \quad (7)$$

where $L(W; x^j, y^j) = [1 - \langle W, \Delta\Phi^j \rangle]_+$, $\Delta\Phi^j \triangleq \Phi(x^j, y^j) - \Phi(x^j, \bar{y}^j)$, and $\bar{y}^j = h(x^j)$. Let $\Omega_k = \{W \in \mathsf{R}^{|S|K \times |S|K}, \text{rank}(W) \leq k\}$, and $\mathcal{P}(W)$ be the primal problem with optimal value p^*.

Different from the usual linear SVM, (7) is not convex, but a biconvex problem (Gorski et al., 2007). To solve it, the straightforward method is blockwise coordinate descent. When β is fixed, (7) is a linear SVM with parameter α, and vise versa. The blockwise coordinate descent works by solving the two SVMs alternately (Algorithm 1).

Blockwise coordinate descent has been widely used for bilinear problems (Gorski et al., 2007; Pirsiavash et al., 2009). It suffers the common local optimum problem. Assume that, in an extreme case, $(\alpha^{(1)}, \beta^{(0)})$ has been a local optimum and no further update on β is needed. Then the algorithm only solves a linear model. We develop a new algorithm which solve α, β simultaneously. It incrementally increases the dual function of (7). The idea is builds on (Shalev-Shwartz & Singer, 2006) and (Shalev-Shwartz & Kakade, 2008), but now the problem is non-convex and more work is needed to compute the gradient and bound the regret. Furthermore, solving from the dual space also provides some insights on the learning problem.

Let $F_k(W) = \frac{1}{2}\|W\|_{\text{F}}^2$ with domain Ω_k, and $\Theta_t = \sum_{j=1}^t \eta_j \Delta\Phi^j$. The dual function of (7) is

$$\mathcal{D}(\eta) = \sum_{j=1}^N \eta_j - \max_{W \in \Omega_1} \left(\langle W, \sum_{j=1}^N \eta_j \Delta\Phi^j \rangle - \frac{1}{2}\|W\|_{\text{F}}^2 \right)$$
$$= \sum_{j=1}^N \eta_j - F_1^*(\Theta_N), \text{ where } \eta_j \in [0, C].$$

Denote $\mathcal{D}_t(\eta_1, \ldots, \eta_{t-1}) = \sum_{j=1}^{t-1} \eta_j - F_1^*(\Theta_{t-1})$. Then $\mathcal{D}(\eta) = \mathcal{D}_{N+1}(\eta_1, \ldots, \eta_N)$.

Proposition 1. $F_1^*(\Theta) = \frac{1}{2}\|\Theta\|_2^2 = \frac{1}{2}\|\Theta\|_{\mathbf{s}(\infty)}^2$.

Proof. Let $\sum_i \sigma_i u_i v_i^\intercal$ be the SVD of Θ.

$$\begin{aligned} F_1^*(\Theta) &= \max_{W\in\Omega_1} \langle W, \Theta\rangle - \frac{1}{2}\|W\|_{\mathrm{F}}^2 \\ &= -\min_{W\in\Omega_1} \frac{1}{2}\|\Theta - W\|_{\mathrm{F}}^2 + \frac{1}{2}\|\Theta\|_{\mathrm{F}}^2 \\ &= -\frac{1}{2}\|\Theta - \sigma_1 u_1 v_1^\intercal\|_{\mathrm{F}}^2 + \frac{1}{2}\|\Theta\|_{\mathrm{F}}^2 \\ &= \frac{1}{2}\sigma_1^2 = \frac{1}{2}\|\Theta\|_2^2. \end{aligned}$$

The third equality is by Eckart-Young-Mirsky theorem. □

Similar to the online mirror descent, at round t ($1 \le t \le T$), our online learning algorithm runs as follows [3]:

- uses $W_{t-1} = \alpha_{t-1}\beta_{t-1}^\intercal$ to predict x^t, $\bar{y}^t = h(x^t)$;
- sets the dual variable η_t as

$$\eta_t = \begin{cases} 0 & \bar{y}^t = y^t \\ C & \bar{y}^t \neq y^t \end{cases}; \tag{8}$$

- updates W_t: $W_t = \nabla F_1^*(\Theta_t) = \alpha_t\beta_t^\intercal$.

Note that we need to compute the gradient of F_1^*. In general, when $p = 1$ or ∞, the Schatten p-norm is not differentiable if there are singular values with multiplicity greater than 1. The following proposition from Watson (1992) will help to compute W_t (see Theorem 2 and Example 1 therein).

Proposition 2. *Let Θ have SVD $\sum_i \sigma_i u_i v_i^\intercal$. If $\sigma_1 \neq \sigma_2$, then F_1^* is differentiable at Θ, and $\nabla F_1^*(\Theta) = \sigma_1 u_1 v_1^\intercal$.*

4.2. The Power Iteration

Updates of α_t, β_t need u_1, v_1 of Θ_t. We use the power iteration to compute them. Roughly, for a matrix Θ and an initial value $\alpha^{(0)}$, if $\sigma_1(\Theta) \neq \sigma_2(\Theta)$, the sequence $\alpha^{(\tau+1)} = \Theta^\intercal\Theta\alpha^{(\tau)}$ (with normalization $\alpha^{(\tau+1)}/\|\alpha^{(\tau+1)}\|$) will converge to u_1. Similarly, $\beta^{(\tau+1)} = \Theta\Theta^\intercal\beta^{(\tau)}$ will converge to v_1. If $\beta^{(0)}$ is set to $\Theta\alpha^{(0)}$, we can also compute u_1, v_1 at the same time: $\alpha^{(\tau+1)} = \Theta\beta^{(\tau)}$, $\beta^{(\tau+1)} = \Theta^\intercal\alpha^{(\tau+1)}$. The convergence speed is determined by $\frac{\sigma_2(\Theta)}{\sigma_1(\Theta)}$ and $\alpha^{(0)}, \beta^{(0)}$ (see Golub & Van Loan (1996), Theorem 8.2.1).

The initial value and normalization are two important components of the power iteration. The former relates to convergence speed, and the latter affects the numerical stability. In the case of Θ_t, if the feature matrices Φ are sparse (e.g., the sequential labelling example), instead of choosing a random initial value and normalizing to the unit ball, we can have better strategies.

For the initial value, recalling that $\Theta_t = \Theta_{t-1} + C\Delta\Phi^t$, one would expect that α_t is close to α_{t-1} if $C\Delta\Phi^t$ is not a "big" change. In fact, Wedin sin theorem (Demmel, 1997) states that $\sin\theta$ is bounded by $\|C\Delta\Phi^t\|$, where $\theta \in [0, \pi/2]$ is the angle between α_{t-1} and α_t. Thus, $\alpha_{t-1}, \beta_{t-1}$ could be good initial values.

For normalization, when α_t, β_t are dense vectors in a high dimensional space, normalization $\alpha^{(\tau+1)}/\|\alpha^{(\tau+1)}\|$ will be time-consuming. Note that only directions of singular vectors are important, and if the feature Φ is sparse, we can use an alternative normalization to speedup. At round t, assume $\|\alpha_{t-1}\| = \|\beta_{t-1}\| = 1$. Let's consider the first multiplication of the power iteration with initial value β_{t-1}: $\alpha_t^{(1)} = (\Theta_{t-1} + C\Delta\Phi^t)\beta_{t-1} = \sigma_1\alpha_{t-1} + C\Delta\Phi^t\beta_{t-1}$, where $\sigma_1 = \sigma_1(\Theta_{t-1})$. We normalize $\alpha_t^{(1)}$ by dividing σ_1:

$$\bar{\alpha}_t^{(1)} = \alpha_{t-1} + \underbrace{\frac{1}{\sigma_1} C\Delta\Phi^t\beta_{t-1}}_{\Delta\alpha^{(1)}}.$$

After the second multiplication, $\beta_t^{(1)} = (\Theta_{t-1} + C\Delta\Phi^t)^\intercal\bar{\alpha}_t^{(1)} = \sigma_1\beta_{t-1} + C(\Delta\Phi^t)^\intercal\alpha_{t-1} + \Theta_t^\intercal\Delta\alpha^{(1)}$. Again, we normalize $\beta_t^{(1)}$ by σ_1:

$$\bar{\beta}_t^{(1)} = \beta_{t-1} + \underbrace{\frac{1}{\sigma_1}\left(C(\Delta\Phi^t)^\intercal\alpha_{t-1} + \Theta_t^\intercal\Delta\alpha^{(1)}\right)}_{\Delta\beta^{(1)}}.$$

We can write the general update equations:

$$\Delta\alpha^{(\tau)} = \frac{1}{\sigma_1}\left(C(\Delta\Phi^t)\beta_{t-1} + \Theta_t\Delta\beta^{(\tau-1)}\right), \tag{9}$$

$$\bar{\alpha}_t^{(\tau)} = \alpha_{t-1} + \Delta\alpha^{(\tau)}, \tag{10}$$

$$\Delta\beta^{(\tau)} = \frac{1}{\sigma_1}\left(C(\Delta\Phi^t)^\intercal\alpha_{t-1} + \Theta_t^\intercal\Delta\alpha^{(\tau)}\right), \tag{11}$$

$$\bar{\beta}_t^{(\tau)} = \beta_{t-1} + \Delta\beta^{(\tau)}. \tag{12}$$

In (9-12), we only update $\Delta\alpha, \Delta\beta$. If $\Delta\Phi$ is a sparse matrix, $\Delta\alpha, \Delta\beta$ are also sparse. Thus, instead of visiting every entry of $\alpha_{t-1}, \beta_{t-1}$, we update entries in $\Delta\alpha, \Delta\beta$, which is more efficient. When the power method converges after R iterations (in experiments, $R = 4$ is enough to converge), we set $\alpha_t = \alpha_t^{(R)}, \beta_t = \beta_t^{(R)}$.

In a word, with the carefully selected initial value and normalization method, the power iteration is an efficient procedure, which only manipulates sparse matrices. The algorithm is summarized in Algorithm 2 [4].

[3] Rigorously, dual variables are $\eta_{t,y}$, where y is any possible output sequence of x^t. Here we set $\eta_t \triangleq \eta_{t,\bar{y}^t}$. Also in (8), we set $\eta_{t,y} = 0$ for $y \neq \bar{y}^t, y^t$.

[4] More implementation details are in the supplementary.

Algorithm 2 Online Learning of the Bilinear Model

Training set:$\{x^j, y^j\}_{j=1}^N$
number of iterations: T, model parameter: C, number of power iterations: R

$\alpha = \frac{\mathbf{1}}{\|\mathbf{1}\|}, \beta = \frac{\mathbf{1}}{\|\mathbf{1}\|}$
for $t = 0$ to T **do**
  for $j = 0$ to N **do**
    $\bar{y}^j = \arg\max_{y \in Y} \alpha^\intercal \Phi(x^j, y)\beta$
    if $\bar{y}^j \neq y^j$ **then**
      $\Theta_t = \Theta_{t-1} + C\Delta\Phi^t$
      $\Delta\alpha^{(0)} = \Delta\beta^{(0)} = \mathbf{0}$ //Power Iteration
      for $\tau = 0$ to R **do**
        $\bar{\alpha}^{(\tau)} = \alpha + \Delta\alpha^{(\tau)}$, by Eq. (9)
        $\bar{\beta}^{(\tau)} = \beta + \Delta\beta^{(\tau)}$, by Eq. (12)
      end for
      $\alpha = \frac{\bar{\alpha}^{(R)}}{\|\bar{\alpha}^{(R)}\|}, \beta = \frac{\bar{\beta}^{(R)}}{\|\bar{\beta}^{(R)}\|}$ $//W_t = \nabla F_1^*(\Theta_t)$
    end if
  end for
end for
return α, β

4.3. Extensions

We give two extensions for the bilinear model and the online learning algorithm.

First, instead of using the bilinear model alone, we can easily incorporate linear models for the 0 order features

$$h(x) = \arg\max_{y \in Y} w^\intercal \hat{\Phi}(x, y) + \alpha^\intercal \Phi(x, y)\beta,$$

where $\hat{\Phi}(x, y) = \sum_i \hat{\varphi}(x, y_i)$ is the 0 order feature vector. The learning algorithm could also be modified correspondingly. The dual objective $\mathcal{D}(\eta)$ now becomes:

$$\sum_{j=1}^N \eta_j - F_1^*(\Theta_N) - \max_w (\langle w, \sum_{j=1}^N \eta_j \Delta\hat{\Phi}^j \rangle - \frac{1}{2}\|w\|^2).$$

For α, β, Algorithm 2 is unchanged, and for w, we have $w_t = w_{t-1} + C\Delta\hat{\Phi}(x^t, y^t)$.

The second extension is about averaging parameters. In previous works on online convex optimization, averaging is a simple method for online batch conversion, and the averaged parameter usually performs better. However, due to the non-convexity of the bilinear formulation, the averaged W may not be in Ω_1 (the sum of rank 1 matrices may not have rank 1). Heuristically, instead of directly averaging W, we can average α and β individually.

5. Analyses

With loss function L_t at round t (the hinge loss here), the regret of an online game against a given strategy U is

$$R_N(U) = \frac{1}{N}\sum_{t=1}^N L_t(W_t) - \frac{1}{N}\sum_{t=1}^N L_t(U).$$

To analyze the regret, we will generally follow the analysis of dual coordinate ascent algorithms. Before starting, it is worth pointing out that, instead of analyzing the dual problem, the online mirror descent framework has provided uniform regret and generalization results for various online learning algorithms (Kakade et al., 2012). However, our bilinear model does not belong to this family. In fact, $F_1(W) = \frac{1}{2}\|W\|_{\mathrm{F}}^2$ is no longer a convex function under the rank constraint, and $F_1^{**} = \frac{1}{2}\|W\|_2^2 \neq F_1$.

Denote $\Delta_t = \mathcal{D}_{t+1}(\eta_1, \ldots, \eta_t) - \mathcal{D}_t(\eta_1, \ldots, \eta_{t-1})$. By weak duality, $\mathcal{D}(\eta_1, \ldots, \eta_N) = \sum_{t=1}^N \Delta_t \leq p^*$. We will show that, with η_t in (8), Δ_t is bounded from below. Our discussions will focus on the case $\eta_t = C$ (when $\eta_t = 0$, no update on W_{t-1}). Expand Δ_t as

$$\Delta_t = C - \frac{1}{2}\|\Theta_{t-1} + C\Delta\Phi^t\|_2^2 + \frac{1}{2}\|\Theta_{t-1}\|_2^2.$$

Proposition 3. *Let* $\Theta \in \mathsf{R}^{m \times n}, l = \min(m, n), F_1^*(\Theta) = \frac{1}{2}\|\Theta\|_2^2$. *If* $\sigma_1(\Theta) \neq \sigma_2(\Theta) > 0$, *the following second order approximation holds in a neighborhood of* Θ*:*

$$F_1^*(\Theta + E) \leq F_1^*(\Theta) + \langle \nabla F_1^*(\Theta), E \rangle + \|E\|_{\mathrm{F}}^2 \frac{2l}{1 - \frac{\hat{\sigma}_2}{\hat{\sigma}_1}},$$

where $[\hat{\sigma}_1, \ldots, \hat{\sigma}_l] = \sigma(\hat{\Theta})$, *and* $\hat{\Theta} = \Theta + \theta E, \theta \in (0, 1)$.

The proof is given in Section 7.1. To compare with online mirror descent algorithms, let's consider the widely used result about the strong smoothness of the squared Schatten norm (Ball et al., 1994; Kakade et al., 2012). Namely, for $p \in [2, \infty], \frac{1}{p} + \frac{1}{q} = 1$,

$$\frac{1}{2}\|\Theta + E\|_{\mathbf{s}(p)}^2 \leq \frac{1}{2}\|\Theta\|_{\mathbf{s}(p)}^2 + \langle \nabla\|\Theta\|_{\mathbf{s}(p)}, E \rangle + \frac{\|E\|_{\mathbf{s}(q)}^2}{2(q-1)}.$$

When $p = \infty$ (the case of F_1^*), the bound is trivial. Kakade et al. (2012) approximated this case by using a finite but sufficiently large p; however, the naive computation of gradients becomes expensive (roughly, it needs the full SVD of Θ). Proposition 3 provides a new (local) bound with respect to the Frobenius norm, which helps to derive the forthcoming regret. .

Let $\sigma(\Theta_t) = [\sigma_1^t, \ldots, \sigma_l^t]$, $\hat{\Theta}_t = \Theta_{t-1} + \theta_t C\Delta\Phi^t$ with $\theta_t \in (0, 1)$ and $\sigma(\hat{\Theta}_t) = [\hat{\sigma}_1^t, \ldots, \hat{\sigma}_l^t]$.

Proposition 4 (Regret)**.** *Assume for all* $\Theta = \Theta_{t-1}, E = C\Delta\Phi^t$, *Proposition 3 holds. Then*

$$R_N(U) \leq \frac{1}{2CN}\|U\|_{\mathrm{F}}^2 + \frac{2lC}{N}\sum_{t=1}^N \frac{\|\Delta\Phi^t\|_{\mathrm{F}}^2}{1 - \frac{\hat{\sigma}_2^t}{\hat{\sigma}_1^t}}.$$

Proof. Note that $\langle \nabla F_1^*(\Theta_{t-1}), \Delta\Phi^t \rangle \leq 0$, we have $\Delta_t \geq CL(W_{t-1}; x^t, y^t) - \frac{2lC^2\|\Delta\Phi^t\|_{\mathrm{F}}^2}{1 - \frac{\hat{\sigma}_2^t}{\hat{\sigma}_1^t}}$ by Proposition 3. Summing over t and by weak duality, the proof is complete. □

We can see that $\frac{\sigma_2^t}{\sigma_1^t}$ controls not only the convergence speed of the power iteration, but also the regret of the online learning algorithm: the larger gap, the tighter regret bound. Thus, if rank 1 approximation is reasonable for our problem, which means σ_1 dominates other singular values, the proposed algorithm is expected to be efficient.

Next, we continue to give a concrete bound of $\frac{\hat{\sigma}_2^t}{\hat{\sigma}_1^t}$. We will make a separable assumption on samples. Define that a matrix W has margin γ with respect to a norm $\|\cdot\|$ if $\min_j \left[\langle \frac{W}{\|W\|}, \Delta\Phi^j \rangle\right] \geq \gamma$.

Proposition 5. *Assume that* $\sup_{j,W} \|\Delta\Phi^j\|_2 \leq M_1$, $\sup_{j,W} \|\Delta\Phi^j\|_{\mathbf{k}(2)} \leq M_2$. *If* $M_1 > \frac{M_2}{2}$ *and* $\exists \tilde{W}$ *has margin* γ *w.r.t.* $\|\cdot\|_{\mathbf{s}(1)}$, *where* $\gamma \in (\frac{M_2}{2}, M_1)$, *then*

$$\frac{\hat{\sigma}_2^t}{\hat{\sigma}_1^t} \leq \frac{M_2 - \gamma}{\gamma}.$$

The proof is given in Section 7.2. Combining Proposition 4 and 5 and noting that $\|\Delta\Phi^t\|_{\mathrm{F}} \leq \sqrt{l}\|\Delta\Phi^t\|_2$, we have the following corollary.

Corollary 6. *Assume the conditions in Propositions 4 and 5 hold, the regret is bounded by*

$$R_N(U) \leq \frac{1}{2CN}\|U\|_{\mathrm{F}}^2 + 2Cl^2M_1^2\frac{\gamma}{2\gamma - M_2}.$$

Let's have a closer look at the two conditions of Proposition 5. First, a sufficient condition for $M_1 > \frac{M_2}{2}$ is that there exits $\delta > 0$, such that for every j, W, $\sigma_1(\Delta\Phi^j) - \sigma_2(\Delta\Phi^j) > \delta$. In other words, σ_1 is "uniformly" greater than σ_2 on the input space. Second, it is clear that $\tilde{W}$ exists if and only if the following trace norm minimization problem has a solution $\|\tilde{W}\|_{\mathbf{s}(1)} < \frac{2}{M_2}$.

$$\min . \frac{1}{2}\|W\|_{\mathbf{s}(1)}^2 \quad \text{s.t. } \langle W, \Delta\Phi^j \rangle \geq 1, \quad \forall j.$$

Previous works on low rank constrained problems usually use the trace norm regularization as an approximation of rank constraints. Similarly, Corollary 6 says that if the problem is well-formed in the trace norm regularization situation, our algorithm which deals with the hard rank constraint will have a small regret bound.

6. Experiments

We present experiments on two sequential labelling tasks (word segmentation and chunking) for which the formulation in Section 3.3 is used.

6.1. Chinese Word Segmentation (CWS)

Given an input sentence in Chinese (a sequence of characters), CWS systems will output a sequence of words by grouping its characters. The data set is from the second SIGHAN Backoff (Emerson, 2005).

We use standard features (Sun et al., 2009; Sun, 2010). Given the current position i, the templates are words at $i-2, i-1, i, i+1, i+2$ and bigrams of them. All of them are 1st order features, and we don't average the model parameters. The performance is measured by the F1-value. We use "BIES" to encode segmentation results, "BIE"represent beginning, inside, and end of a word, and "S" is a word with only on character. Thus, feature numbers of the bilinear model (10^7) are only 50% of the linear model.

The algorithms for comparison are [5]: "bol" which is the proposed method with $T = 20, C = 1, R = 4$, "bcd" which is the blockwise coordinate descent with SVM solver in Shalev-Shwartz & Singer (2006) ($C = 1$). We also compare state-of-the-art online linear model "sp" which is the structured perceptron with $T = 20$, learning rate $C = 1$ (note that "sp" can be seen as a solver of the structured SVM (Freund & Schapire, 1999)).

Figure 1 describes the performances with different training data sizes [6]. In general, our method is the best on pku, cityu, and as. Especially, when the training set is small, the advantage of "bol" is more obvious. It suggests that, compared with "sp", the prior knowledge on feature functions will be helpful if we are lack of training data. At the same time, compared with "bol", the new learning algorithm could prevent the training process being attracted by a solution near the 0-order model which is less expressive.

In Table 1, we further compare the performances of online learning algorithms with models learned by conditional random fields (CRF) [7]. The "crf2" is one of the most powerful batch learner for sequential labelling, and "crf1" enforce the sparsity on CRF models. We also lists the training time on data set as (other data sets are similar). It shows that "bol" is competitive to "crf2" on pku and msr, and outperforms "crf1" on pku, msr and cityu. But the CRFs need more time for training (Figure 2).

For the power iteration, we examined its convergence by sampling at different rounds on the dev set. The conclusion is that a small R (≤ 4) is sufficient to converge. Thus, for this particular problem, the singular vectors of Θ_t are actually close to those of Θ_{t-1}, and the power iteration is efficient with the initial value we have chosen. Note that, even if the singular vectors change severely, our initial value still helps to avoid manipulations on dense vectors, which may also speed up the iteration.

[5] T and C are from Sun (2010), and the value R is selected on a dev set (10% of the pku training set).

[6] We define that one method is better than another if it is better on 8 points (out of 10) at least.

[7] http://crfpp.googlecode.com/

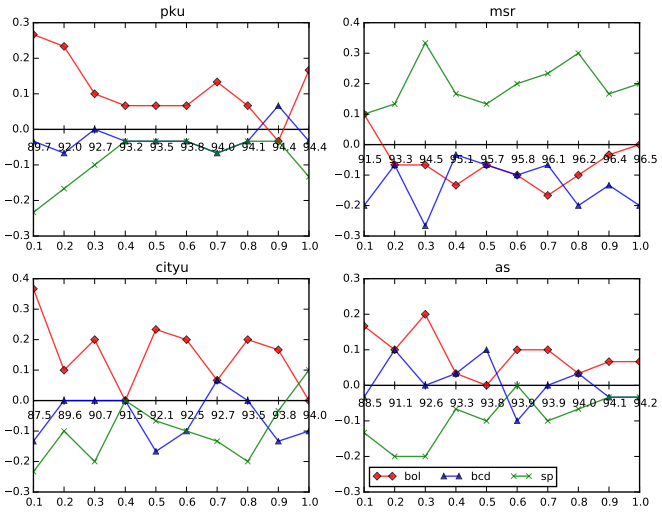

Figure 1. Performances on word segmentation. x-axis is the proportion of training set. The horizontal line $y = 0$ is the average F1 value among algorithms, and y-axis is the offset against average.

Models	F1				Training Time
	pku	msr	cityu	as	$\times 10^3$s
bol	94.6	96.5	94.0	94.3	2.4
bcd	94.4	96.3	93.9	94.2	2.4
sp	94.3	96.7	94.1	94.2	1.5
crf2	93.3	96.5	94.2	94.6	9.8
crf1	92.5	96.1	93.5	95.0	8.3

Table 1. Comparison with batch learners. "crf2" is the CRF with L2 regularization, and "crf1" is the CRF with L1 regularization.

6.2. Text Chunking

The task of text chunking divides a sentence in syntactically correlated parts of words (e.g., noun phrase, verb phrase). We conduct experiments on the CoNLL Shared-task 2000 (Sang & Buchholz, 2000). Different from the word segmentation task, in order to show the extendability of our model and algorithm, we include both the 1 order and the 0 order features, and average the parameters. The tag set size is 23 (e.g., B-NP, I-NP, B-VP, I-VP). The results (Figure 3) are similar to the task of word segmentation in general, which means that the two extensions to the basic bilinear model are effective.

7. Proofs

7.1. The Proof of Proposition 3

The following expression of a Hessian matrix is fundamental for our analysis. ((Overton & Womersley, 1995), more general results are given in (Lewis & Sendov, 2001))

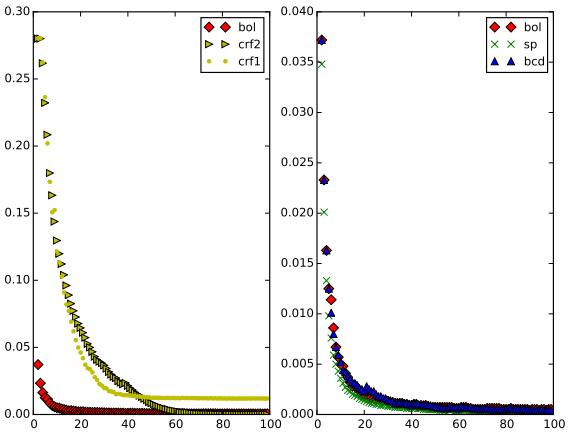

Figure 2. Convergence speeds on the pku data set. x-axis is the number of iterations, y-axis is the label error rate. "bol" converges much faster than CRFs, and it is slightly slower than "sp".

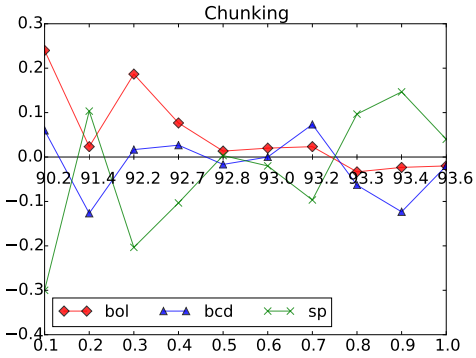

Figure 3. Performances on chunking. x-axis represents the proportion of training set. y-axis is the F1-value.

Let $A(a)$ be an $n \times n$ symmetric matrix-valued function, $a \in \mathsf{R}^m$. Assume $A(a)$ is twice continuously differentiable, with the second derivative satisfying a Lipschitz condition on a. The eigenvalues of $A(a)$ are $\lambda_1(a) \geq \cdots \geq \lambda_n(a)$.

Proposition 7. *Let $A(a)$ have eigen decomposition at $\hat{a}$*

$$A(\hat{a}) = \hat{Q}\mathrm{Diag}(\lambda_1(\hat{a}), \ldots, \lambda_n(\hat{a}))\hat{Q}^\intercal,$$

where $\hat{Q} = [\hat{q_1}, \hat{q_2}, \ldots, \hat{q_n}]$, $\hat{Q}^\intercal\hat{Q} = I$. If $\lambda(\hat{a})$ are distinct, then the second derivative of $\lambda_d(a)$ $(1 \leq d \leq n)$ at $\hat{a}$ is

$$\frac{\partial^2\lambda_d(\hat{a})}{\partial a_x \partial a_u} = \hat{q}_d^\intercal \frac{\partial^2 A(\hat{a})}{\partial a_x \partial a_u}\hat{q}_d + 2\sum_{s\neq d}\frac{\hat{q}_d^\intercal \frac{\partial A(\hat{a})}{\partial a_x}\hat{q}_s \hat{q}_d^\intercal \frac{\partial A(\hat{a})}{\partial a_u}\hat{q}_s}{\lambda_d - \lambda_s}.$$

Given a matrix $B \in \mathsf{R}^{m\times n}$, B can be seen as a function of $\mathrm{vec}(B)$. $\sigma_d(B) = \sqrt{\lambda_d(B^\intercal B)}$. We have the following corollary of Proposition 7.

Corollary 8. *Assume that $B \in \mathsf{R}^{m\times n}$, $b = \mathrm{vec}(B)$, and B*

has singular value decomposition $P\Sigma Q^\intercal$, where

$$P=[p_1,p_2,\ldots,p_m], Q=[q_1,q_2,\ldots,q_n]$$
$$\Sigma=\mathrm{Diag}(\sigma_1,\ldots,\sigma_l), l=\min(m,n).$$

If $\sigma(B)$ are distinct, then the Hessian matrix of σ_d w.r.t. b is $C+\frac{1}{\sigma_d}D$, where $C\preceq 0$ and D has entry $D_{xy,uv}$:

$$q_d^y q_d^v \mathbb{I}(x=u)+\sum_{s\neq d}\frac{(q_d^y p_s^x\sigma_s+q_s^y p_d^x\sigma_d)(q_d^v p_s^u\sigma_d+q_s^v p_d^u\sigma_s)}{\sigma_d^2-\sigma_s^2},$$

where xy, uv are indices of b, and p^y is the y-th entry of p.

Proof. Denote $A=B^\intercal B$, $A=(a_{ij}), B=(b_{xy})$. By the chain rule

$$\frac{\mathrm{d}\sigma_d}{\mathrm{d}b}=\frac{\mathrm{d}\sigma_d}{\mathrm{d}\lambda_d(A)}\frac{\mathrm{d}\lambda_d(A)}{\mathrm{d}b}=\frac{1}{2\sigma_d}\frac{\mathrm{d}\lambda_d(A)}{\mathrm{d}b},$$
$$\frac{\mathrm{d}^2\sigma_d}{\mathrm{d}b^2}=\underbrace{-\frac{1}{4\sigma_d^3}\frac{\mathrm{d}\lambda_d(A)}{\mathrm{d}b}\otimes\frac{\mathrm{d}\lambda_d(A)}{\mathrm{d}b}}_{C}+\frac{1}{\sigma_d}\underbrace{\frac{1}{2}\frac{\mathrm{d}^2\lambda_d(A)}{\mathrm{d}b^2}}_{D}.$$

Note that $a_{ij}=\sum_k b_{ki}b_{kj}$,

$$\left(\frac{\partial A}{\partial b_{xy}}\right)_{ij}=\begin{cases}b_{xj} & i=y, j\neq y\\ b_{xi} & i\neq y, j=y\\ 2b_{xy} & i=y, j=y\\ 0 & i\neq y, j\neq y\end{cases}.$$

We have

$$q_d^\intercal\frac{\partial A}{\partial b_{xy}}q_s=\sum_i b_{xi}(q_d^y q_s^i+q_d^i q_s^y)=q_d^y p_s^x\sigma_s+q_s^y p_d^x\sigma_d.$$

Similarly,

$$q_d^\intercal\frac{\partial A}{\partial b_{xy}\partial b_{uv}}q_d=2q_d^y q_d^v\mathbb{I}(x=u).$$

Using Proposition 7 completes the proof. □

For a matrix A, let $\mathrm{vec}(A)=[a_1^\intercal,a_2^\intercal,\ldots,a_n^\intercal]^\intercal$, where a_i are columns of A. The following is the proof of Proposition 3.

Proof. Assume w.l.o.g. $l=m$. The Taylor expansion (with the Lagrange remainder) of F_1^* at Θ is

$$F_1^*(\Theta+E)=F_1^*(\Theta)+\langle\nabla F_1^*(\Theta),E\rangle+\mathrm{vec}(E)^\intercal H(\hat{\Theta})\mathrm{vec}(E),$$

where H is the Hessian matrix. The aim is to bound the remainder. By the chain rule and Corollary 8 with $d=1$,

$$\frac{\mathrm{d}^2F_1^*}{\mathrm{d}\Theta^2}=\frac{\mathrm{d}\sigma_1}{\mathrm{d}\Theta}\otimes\frac{\mathrm{d}\sigma_1}{\mathrm{d}\Theta}+\|\Theta\|_2 C+D.$$

The convexity of $\|\Theta\|_2$ implies that $D\succeq 0$. Since D is also symmetric, we have $\|D\|_2\leq\mathrm{Tr}(D)$. It is easy to show that

$$\mathrm{Tr}(D)=l+\sum_{s=2}^{l}\frac{\sigma_1^2+\sigma_s^2}{\sigma_1^2-\sigma_s^2}\leq l+\sum_{s=2}^{l}\frac{\sigma_1+\sigma_s}{\sigma_1-\sigma_s}$$
$$\leq l+(l-1)\left(1+\frac{2}{\frac{\sigma_1}{\sigma_2}-1}\right).$$

By Theorem 2 of Watson (1992),

$$\mathrm{vec}(E)^\intercal H(\hat{\Theta})\mathrm{vec}(E)\leq\|E\|_{\mathrm{F}}^2\left(\|\frac{\mathrm{d}\sigma_1}{\mathrm{d}\Theta}\otimes\frac{\mathrm{d}\sigma_1}{\mathrm{d}\Theta}\|_2+\|D\|_2\right)\Big|_{\hat{\Theta}}$$
$$\leq\|E\|_{\mathrm{F}}^2\frac{2l}{1-\frac{\hat{\sigma}_2}{\hat{\sigma}_1}}.$$ □

7.2. The Proof of Proposition 5

Proof. We first give a lower bound on $\hat{\sigma}_1^t$. By von Neumann's inequality (Bhatia, 1997), for any W,Θ,

$$\langle W,\Theta\rangle\leq\langle\sigma(W),\sigma(\Theta)\rangle\leq\|W\|_{\mathbf{s}(1)}\|\Theta\|_{\mathbf{s}(\infty)}.$$

Let $W=\tilde{W},\Theta=\hat{\Theta}_t$. We have

$$\hat{\sigma}_1^t\geq\frac{1}{\|\tilde{W}\|_{\mathbf{s}(1)}}\langle\tilde{W},\hat{\Theta}_t\rangle=\frac{C}{\|\tilde{W}\|_{\mathbf{s}(1)}}\langle\tilde{W},\sum_{j=1}^{t-1}\Delta\Phi^j+\theta_t\Delta\Phi^t\rangle$$
$$\geq(t-1+\theta_t)C\gamma.$$

Then,

$$1+\frac{\hat{\sigma}_2^t}{\hat{\sigma}_1^t}=\frac{\|\hat{\Theta}_t\|_{\mathbf{k}(2)}}{\hat{\sigma}_1^t}\leq\frac{(t-1+\theta_t)CM_2}{\hat{\sigma}_1^t}\leq\frac{M_2}{\gamma}.$$

Rearranging the equation leads to the result. We remark that $\gamma>\frac{M_2}{2}$ is necessary for a non-trivial bound; on the other hand, $\tilde{W}$ has margin γ implies that

$$\gamma\leq\frac{1}{\|\tilde{W}\|_{\mathbf{s}(1)}}\langle\tilde{W},\Delta\Phi^j\rangle\leq\|\Delta\Phi^j\|_2\leq M_1.$$ □

8. Conclusion

We presented a bilinear model with matrix features. A simple online algorithm was derived and analyzed. Empirical results on sequential labelling tasks showed that the proposed method is competitive and efficient. In future work, it is interesting to explore models with $\mathrm{rank}\leq k$ (if $k>1$, we need an efficient algorithm to compute ∇F_k^*, which is roughly the first k singular vectors). And it is meaningful to bound $\frac{\sigma_2}{\sigma_1}$ under other conditions.

Acknowledgments

The authors wish to thank the reviewers for their helpful comments and suggestions. This research is supported by NSFC (61402175, 61370175) and IIPL-2014-008.

References

Ball, Keith, Carlen, Eric A., and Lieb, Elliott H. Sharp uniform convexity and smoothness inequalities for trace norms. *Inventiones mathematicae*, 115:463–482, 1994.

Bhatia, Rajendra. *Matrix Analysis*. Springer New York, 1997.

Demmel, James W. *Applied Numerical Linear Algebra*. Society for Industrial and Applied Mathematics, 1997.

Duchi, John C., Shalev-Shwartz, Shai, Singer, Yoram, and Tewari, Ambuj. Composite objective mirror descent. In *COLT*, pp. 14–26, 2010.

Emerson, Thomas. The second international Chinese word segmentation bakeoff. In *the Second SIGHAN Workshop on Chinese Language Processing*, pp. 123 – 133, 2005.

Freund, Yoav and Schapire, Robert E. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296, 1999.

Golub, Gene H. and Van Loan, Charles F. *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, 1996.

Gorski, Jochen, Pfeuffer, Frank, and Klamroth, Kathrin. Biconvex sets and optimization with biconvex functions: a survey and extensions. *Mathematical Methods of Operations Research*, 66:373–407, 2007.

Kakade, Sham M., Shalev-Shwartz, Shai, and Tewari, Ambuj. Regularization techniques for learning with matrices. *JMLR*, 13:1865–1890, 2012.

Lei, Tao, Xin, Yu, Zhang, Yuan, Barzilay, Regina, and Jaakkola, Tommi. Low-rank tensors for scoring dependency structures. In *ACL*, pp. 1381–1391, 2014.

Lewis, Adrian S. and Sendov, Hristo S. Twice differentiable spectral functions. *SIAM Journal on Matrix Analysis and Applications*, 23:368–386, 2001.

Overton, Michael L. and Womersley, Robert S. Second derivatives for optimizing eigenvalues of symmetric matrices. *SIAM Journal on Matrix Analysis and Applications*, 16:697–718, 1995.

Pirsiavash, Hamed, Ramanan, Deva, and Fowlkes, Charless. Bilinear classifiers for visual recognition. In *NIPS*, 2009.

Rendle, Steffen. Factorization machines. In *ICDM*, pp. 995–1000, 2010.

Rennie, Jason D. M. and Srebro, Nathan. Fast maximum margin matrix factorization for collaborative prediction. In *ICML*, pp. 713–719, 2005.

Sang, Erik F. Tjong Kim and Buchholz, Sabine. Introduction to the CoNLL-2000 shared task: Chunking. In *Proc. of CoNLL and LLL*, 2000.

Shalev-Shwartz, Shai. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2012.

Shalev-Shwartz, Shai and Kakade, Sham M. Mind the duality gap: Logarithmic regret algorithms for online optimization. In *NIPS*, pp. 1457–1464, 2008.

Shalev-Shwartz, Shai and Singer, Yoram. Online learning meets optimization in the dual. In *COLT*, pp. 423–437, 2006.

Shalev-Shwartz, Shai and Zhang, Tong. Stochastic dual coordinate ascent methods for regularized loss. *JMLR*, 14:567–599, 2013.

Shalev-Shwartz, Shai, Gonen, Alon, and Shamir, Ohad. Large-scale convex minimization with a low-rank constraint. In *ICML*, pp. 329–336, 2011.

Srebro, Nathan, Rennie, Jason D. M., and Jaakola, Tommi S. Maximum-margin matrix factorization. In *NIPS*, pp. 1329–1336, 2005.

Sun, Weiwei. Word-based and character-based word segmentation models: Comparison and combination. In *Coling: Posters*, pp. 1211–1219, 2010.

Sun, Xu, Zhang, Yaozhong, Matsuzaki, Takuya, Tsuruoka, Yoshimasa, and Tsujii, Jun'ichi. A discriminative latent variable Chinese segmenter with hybrid word/character information. In *NAACL*, pp. 56–64, 2009.

Tenenbaum, Joshua B. and Freeman, William T. Separating style and content with bilinear models. *Neural Computation*, 12:1247–1283, 2000.

Wang, Jialei, Hoi, Steven C.H., Zhao, Peilin, and Liu, Zhi-Yong. Online multi-task collaborative filtering for on-the-fly recommender systems. In *Proceedings of the 7th ACM Conference on Recommender Systems*, pp. 237–244, 2013.

Watson, G.A. Characterization of the subdifferential of some matrix norms. *Linear Algebra and its Applications*, 170:33 – 45, 1992.