Training Conditional Random Fields by Periodic Step Size Adaptation for Large-Scale Text Mining

Han-Shen Huang Yu-Ming Chang Chun-Nan Hsu
Institute of Information Science
Academia Sinica
Taipei, Taiwan
{hanshen,porter,chunnan}@iis.sinica.edu.tw

Abstract

For applications with consecutive incoming training examples, on-line learning has the potential to achieve a likelihood as high as off-line learning without scanning all available training examples and usually has a much smaller memory footprint. To train CRFs on-line, this paper presents the Periodic Step size Adaptation *(PSA) method to dynamically adjust the learning rates in stochastic gradient descent. We applied our method to three large scale text mining tasks. Experimental results show that PSA outperforms the best off-line algorithm, L-BFGS, by many hundred times, and outperforms the best on-line algorithm, SMD, by an order of magnitude in terms of the number of passes required to scan the training data set.*

1. Introduction

Many algorithms have been proposed to train Conditional Random Fields (CRF). Among them, the limited-memory BFGS (L-BFGS) algorithm [12] has become virtually the standard algorithm for training CRF. Though L-BFGS can significantly reduce the number of iterations required, it still needs to scan the training set many times, which is impractical for some large scale applications. On-Line learning has the potential to obtain a parameter vector that achieves a likelihood as high as that achieved by off-line learning even before scanning the entire training set. Another advantage of on-line learning is that it has a much smaller memory footprint than its off-line learning counterparts. Vishwanathan et al. [17] proposed a stochastic meta-descent (SMD) method for CRF training, demonstrating that on-line learning is feasible for large-scale CRF training. However, experimental results show that though SMD can reduce the number of passes required to scan the training set, it requires careful tunings and may sometimes converge to a parameter vector that is worse than that which off-line algorithms can reach.

In this paper, we present the periodic step size adaptation (PSA) method derived by aggregating on-line update mappings. Though there are already many step size adaptation methods available, our method is different in that we adjust the learning rate periodically. Our adjustment is very simple but is based on accurate approximation of optimal adjustment. Periodic adjustment is proved to improve the accuracy of approximation. We applied our method to three large scale text mining tasks. Experimental results show that PSA outperforms the best off-line algorithm, L-BFGS, by many hundred times, and outperforms the best on-line algorithm, SMD, by an order of magnitude.

2. Conditional Random Fields

The CRF [9] is one of the most prevailing solutions to sequential data classification. In a CRF, sequences and their labels are transformed into features. The probability of a labeling result is a function of weighted sum of features. Training of CRFs is to assign proper weights for all features, trying to minimize the negative log-likelihood or penalized negative log-likelihood with the training data.

Let $\mathcal{D} = \{(x_1, y_1), \ldots, (x_T, y_T)\}$ denote a set of T data sequences x_k and the corresponding labels y_k. A CRF defines l features to be transformed from a given instance (x, y): $F(x, y) = (f_1(x, y), \ldots, f_l(x, y))^T$, where $f_i(x, y)$ is the number of times that feature i occurs in (x, y). A CRF is parameterized by the weights for all features: $\theta = (\theta_1, \ldots, \theta_l)^T$. Then, the conditional probability of y given x is: $p_\theta(y|x) = \frac{\exp(\theta^T F(x,y))}{Z_\theta(x)}$, where $Z_\theta(x)$ is a normalization term: $Z_\theta(x) = \sum_y \exp(\theta^T F(x, y))$. Training of CRFs is to search for the weight vector that minimizes the negative log-likelihood function as the objective function, which, denoted by $L_D(\theta)$, is: $L_\mathcal{D}(\theta) = -\sum_k \log p_\theta(y_k|x_k) = -\sum_k \left[\theta^T F(x_k, y_k) - \log Z_\theta(x_k)\right]$.

3. Training CRF Off-line

The generalized iterative scaling (GIS) algorithm [2], originally designed to train maximum entropy models, became well-known because of its use as the training algorithm for CRF [9]. Since GIS is extremely slow to converge, other approaches to CRF training that applies gradient-based numerical optimization algorithms have been proposed [10]. Among them, the limited memory variable metrics (L-BFGS) method has become the de facto standard now. L-BFGS is a second-order method that is derived from the second-order Taylor expansion: $\mathcal{L}_{\mathcal{D}}(\theta+\Delta) \approx \mathcal{L}_{\mathcal{D}}(\theta) + \Delta^T g(\theta) + \frac{1}{2}\Delta^T \mathbf{H}(\theta)\Delta$, where $g(\theta)$ is $\nabla\mathcal{L}_D(\theta)$, and $\mathbf{H}(\theta)$ is the Hessian matrix of $\mathcal{L}_D(\theta)$ given by $\mathbf{H}(\theta) = \frac{\mathbf{I}}{\sigma^2} + \text{Cov}(\theta)$, where $\text{Cov}(\theta)$ is the covariance of the feature vectors under model distribution $p_\theta(x, y)$. By applying Newton's method, we obtain an update rule as follows:

$$\theta^{(t+1)} = \theta^{(t)} - \mathbf{H}^{-1}(\theta^{(t)})g(\theta^{(t)}). \quad (1)$$

The Hessian matrix of the log-likelihood involves the covariance matrix, which is usually difficult to compute given even a small number of features. L-BFGS overcomes the problem by approximating the $l \times l$ Hessian matrix with a limited memory $c \times l$ *variable metric*, where c is a small constant, in the range of 3 to 10, while l can be millions. A huge saving is therefore achieved. Malouf [10] reported that L-BFGS significantly outperforms GIS and other gradient-based algorithms in terms of the rate of convergence.

4. Accelerating Generalized Iterative Scaling

4.1. Aitken's Acceleration

When applied to large training sets for CRF, GIS is extremely slow to converge. One approach to accelerating GIS is to consider GIS as a fixed-point iteration mapping and apply Aitken's acceleration. Suppose that we apply the fixed-point iteration method from $\theta^{(t)}$ in the neighborhood of θ^* and the iteration converges at θ^*. Assuming that the mapping M is differentiable, we can apply a linear Taylor expansion of M around θ^* so that

$$\begin{aligned} \theta^{(t+1)} &= M(\theta^{(t)}) \\ &\approx \theta^* + \mathbf{J}(\theta^{(t)} - \theta^*), \end{aligned} \quad (2)$$

where $\mathbf{J}$ abbreviates $M'(\theta^*)$, the Jacobian of the mapping M at θ^*. Assuming that $\text{eig}(\mathbf{J}) \in (-1, 1)$, the multivariate Aitken's acceleration is given by [11]:

$$\begin{aligned} \theta^* &\approx \theta^{(t)} + \sum_{h=0}^{\infty} \mathbf{J}^h(\theta^{(t+1)} - \theta^{(t)}) \\ &= \theta^{(t)} + (\mathbf{I} - \mathbf{J})^{-1}(\theta^{(t+1)} - \theta^{(t)}), \end{aligned} \quad (3)$$

.

4.2. Triple Jump Extrapolation

One of the drawbacks of Aitken's acceleration is that it must evaluate the Jacobian, which involves the covariance matrix that may not have a closed form and can be intractable even for a very simple model with a low dimensional parameter space. Other drawbacks include that it may not always converge and that it may be numerically unstable [4]. One may approximate the Jacobian to overcome these drawbacks. For example, replace $(\mathbf{I} - \mathbf{J})^{-1}$ in Equation (3) with a dynamically adjusted scalar [13].

Here we review another method where the adjustment can be guided as follows. Let $\psi^* := Q^{-1}\theta^*$ be the eigen transformed θ^*. Then in the eigenspace of $\mathbf{J}$, we have $\psi_i^* \approx \psi_i^{(t)} + \frac{1}{1-\lambda_i}(\psi_{Mi}^{(t)} - \psi_i^{(t)})$, where λ_i is the i-th eigenvalue of $\mathbf{J}$. When Q^{-1} is not close to $\mathbf{I}$, we can simplify Aitken's acceleration by replacing λ_i with $\gamma^{(t)}$ such that $\gamma^{(t)} \approx \lambda_{max}$ at the t-th iteration: $\theta^{(t+1)} = \theta^{(t)} + (1-\gamma^{(t)})^{-1}(\theta_M^{(t)} - \theta^{(t)})$. Since $\mathbf{J}(\theta^{(t)} - \theta^{(t-1)}) \approx \theta^{(t+1)} - \theta^{(t)} = \theta_M^{(t)} - \theta^{(t)}$, we define

$$\gamma^{(t)} := \frac{\|\theta_M^{(t)} - \theta^{(t)}\|}{\|\theta^{(t)} - \theta^{(t-1)}\|}. \quad (4)$$

Because to extrapolate to $\theta^{(t+1)}$, we need to apply M consecutively to obtain $\theta^{(t)}$ and $\theta_M^{(t)}$, Huang et al. [3] named this method as the *Triple Jump Acceleration*.

It is also possible to extrapolate with a different rate for each component or sub-vector of the parameter vector. Let p be the index of a component or sub-vector in θ. Then with

$$\gamma_p^{(t)} := \frac{\|\theta_{Mp}^{(t)} - \theta_p^{(t)}\|}{\|\theta_p^{(t)} - \theta_p^{(t-1)}\|}, \quad (5)$$

we can perform extrapolation for each component or sub-vector by $\theta_p^{(t+1)} = \theta_p^{(t)} + (1 - \gamma_p^{(t)})^{-1}(\theta_{Mp}^{(t)} - \theta_p^{(t)}), \quad \forall p.$

5. Training CRF On-Line

5.1. Stochastic Gradient Descent Methods

Stochastic gradient descent (SGD) methods iteratively update the parameters of a model with gradients computed by small batches of b examples. Unlike gradient descent, in SGD, the global objective function $\mathcal{L}_{\mathcal{D}}(\theta)$ is not accessible during the stochastic search. Instead, only a local objective function $\mathcal{L}^{(t)}(\theta)$ based on the batch of examples at iteration t is used. Let examples be drawn from $\mathcal{D}$ with equal weights. Then, the following relation exists between $\mathcal{L}^{(t)}(\theta)$ and $\mathcal{L}_{\mathcal{D}}(\theta)$: $\frac{E(\mathcal{L}^{(t)}(\theta))}{b} = \frac{\mathcal{L}_{\mathcal{D}}(\theta)}{|\mathcal{D}|}$, where $E(\mathcal{L}^{(t)}(\theta))$ is the expectation with respect to the distribution of examples in $\mathcal{D}$, and $|\mathcal{D}|$ is the number of examples in $\mathcal{D}$. The above equation describes that the normalized objective

function given $\mathcal{D}$ is the expectation of the normalized on-line objective function given a random batch.

Let $g^{(t)}$ denote $\frac{\partial \mathcal{L}^{(t)}(\theta^{(t)})}{\partial \theta}$. The parameter vector is updated with the following equation by SGD methods:

$$\theta^{(t+1)} = \theta^{(t)} - \eta^{(t)} \cdot g^{(t)}. \quad (6)$$

It has been proved that SGD with a proper fixed small learning rate η can converge to the neighborhood of θ^* [8], but the convergence rate will be extremely slow. SGD can converge faster with a relatively large learning rate, which will be annealed or discounted gradually to ensure convergence. For further acceleration, second-order methods are usually adopted in which estimation of Hessians is required.

5.2. Stochastic Meta Descent

Stochastic meta descent (SMD) [17] is currently the fastest SGD method for training CRF reported in the literature. In CRF, most of the elements of the parameter vector are mutually independent so that the Hessian is close to a diagonal matrix. Therefore, they approximate the Hessian by means of a vector storing the diagonal elements. SMD uses a vector of local learning rates in Equation (6).

Other on-line algorithms that may potentially be applied to the training of CRF include the Margin Infused Relaxed Algorithm (MIRA) [1]. On each on-line update, MIRA attempts to keep current and new parameter vectors as close as possible, subject to correctly classifying the given data with margins to incorrect classifications larger than some loss function. MIRA has been included in the latest release of CRF++ [6].

6. Our Method

6.1. Optimal Step Size Adaptation

In stochastic gradient descent, approximated objective function is used to compute the gradient. $\mathcal{L}(\theta;\mathcal{D}) = \sum_{t=0}^{\frac{|D|}{b}-1}\big(L_B(\theta)+\frac{b\|\theta\|^2}{2|\mathcal{D}|\sigma^2}\big)$, where $B^{(t)}$ is a batch of b examples $\subseteq \mathcal{D}$. In off-line gradient descent, we apply this update rule: $\theta^{(t+1)} = \theta^{(t)} - \eta\nabla\mathcal{L}(\theta^{(t)};\mathcal{D})$, but in on-line situation, the update rule becomes $\theta^{(t+1)} = \theta^{(t)} - \eta\nabla\mathcal{L}(\theta^{(t)};B^{(t)})$.

Using $\frac{\eta}{b}$ as the step size in SGD, we have $\theta^{(t+1)} = \theta^{(t)} - \eta\big(\frac{\nabla L_B}{b} + \frac{\theta^{(t)}}{|\mathcal{D}|\sigma^2}\big) = \theta^{(t)} - \eta g^{(t)} := M_b(\theta^{(t)})$. We can consider that the on-line mapping M_b is a random variable with its mean equal to the off-line mapping, where the entire set of data is used to evaluate the gradient.

From Aitken's acceleration, to reach θ^*, we need $\theta^{(t+1)} = \theta^{(t)} + (\mathbf{I}-\mathbf{J})^{-1}(M(\theta^{(t)})-\theta^{(t)}) \approx \theta^{(t)} - \frac{\eta}{1-\gamma}g^{(t)}$. Therefore, we can update η_i componentwise by

$$\eta_i^{(t+1)} = \eta_i^{(t)}\frac{1}{1-\gamma_i^{(t)}}, \quad (7)$$

where $\gamma_i^{(t)}$ can be estimated by Equation (5). However, due to stochastic nature of the mapping, it is unlikely that we can obtain an accurate eigenvalue estimation at each iteration. Moreover, since Aitken's acceleration does not guarantee convergence, when the extrapolation does not improve the log-likelihood, we can throw away that extrapolation and apply the original mapping to ensure convergence in off-line applications, but in on-line situations, this is impossible. Therefore, we need to derive a new method to use the estimates of the eigenvalues in on-line situations.

6.2. Periodic Step Size Adaptation

We follow the SGD framework in Equation (6) to adjust the learning rates for on-line text mining with huge models. Our method uses a vector of local learning rates corresponding to each dimension of the parameter vector. The strategy is to discount learning rates based on the estimated eigenvalues of each dimension: higher/lower discounts for learning rates of fast/slow dimensions. This strategy is consistent with the optimal step size adaptation given in Equation (7).

In our method, we consider Equation (6) as an on-line mapping $M_b(\theta) : \mathbb{R}^n \mapsto \mathbb{R}^n$: $M_b(\theta^{(t)}) := \theta^{(t)} - \eta\cdot g^{(t)}$. We also assume that $\mathbf{H}^{(t)} = \mathbf{H}(\theta^{(t)})$ is a diagonal matrix. Taking the derivative of $M_b(\theta^{(t)})$ with respect to θ, we can obtain $\mathbf{J}^{(t)}$, the Jacobian of M_b at $\theta^{(t)}$, which is also a diagonal matrix: $\frac{\partial}{\partial\theta}M_b(\theta^{(t)}) := \mathbf{J}^{(t)} = \mathbf{I} - \eta\mathbf{H}^{(t)}$. In on-line gradient descent methods, however, the estimation is usually inaccurate because only small batches of data are used to update parameter vectors. To reduce the deviation to the true eigenvalues, we aggregate consecutive mappings to average n estimations. We use four parameter vectors, $\theta^{(t)}$, $\theta^{(t+1)}$, $\theta^{(t+n)}$, and $\theta^{(t+n+1)}$, to estimate eigenvalues. In an off-line situation, we apply Taylor expansion at θ^* and derive the following relation : $\theta^{t+n+1} - \theta^{t+n} \approx (\mathbf{J}^*)^n(\theta^{t+1} - \theta^t)$. From the result, we can estimate the eigenvalues to the n-th power by $\gamma_i^n := \frac{\theta_i^{(t+n+1)} - \theta_i^{(t+n)}}{\theta_i^{(t+1)} - \theta_i^{(t)}}$.

After running $2n$ iterations, we aggregate these estimates by computing their average, weighted by the ratio of componentwise one step difference and the entire search path along dimension i after n iterations, that is, $\theta_i^{(t+n)} - \theta_i^{(t)}$. This is equivalent to averaging the numerators and denominators separately, resulting in efficient estimation of the average of γ_i^n, denoted by $\bar{\gamma}_i^n$:

$$\begin{aligned}\bar{\gamma}_i^n &:= \frac{\sum_{k=0}^{n-1}\frac{\theta_i^{(t+n+k+1)} - \theta_i^{(t+n+k)}}{n}}{\sum_{k=0}^{n-1}\frac{\theta_i^{(t+k+1)} - \theta_i^{(t+k)}}{n}} \\ &= \frac{\theta_i^{(t+2n)} - \theta_i^{(t+n)}}{\theta_i^{(t+n)} - \theta_i^{(t)}}. \quad (8)\end{aligned}$$

For computational stability, we introduce a constant κ as

the upper bound of $|\bar{\gamma}_i^n|$. Let a_i denote the constrained $\bar{\gamma}_i^n$:

$$a_i := \mathrm{sgn}(\bar{\gamma}_i^n) \min(|\bar{\gamma}_i^n|, \kappa). \quad (9)$$

Instead of updating the learning rate at every iteration, we update the learning rates every $2n$ iterations based on a_i's:

$$\eta_i^{(t+2n+1)} = b_i \eta_i^{(t+2n)}, \quad (10)$$

where

$$b_i = \frac{M + a_i}{M + \kappa + m}. \quad (11)$$

In Equation (11), M and m control the scale of discount factors b_i's. We can define the maximal value α and minimal value β of the discount and then obtain M and m by solving $\beta \leq b_i \leq \alpha$. Since $-\kappa \leq a_i \leq \kappa$, we have $b_i = \alpha$ when $a_i = \kappa$ and $b_i = \beta$ when $a_i = -\kappa$. By solving the equations, M and m are:

$$M = \frac{\alpha + \beta}{\alpha - \beta}\kappa \text{ and } m = \frac{2(1-\alpha)}{\alpha - \beta}\kappa. \quad (12)$$

At last, we summarize our SGD method as follows:

Algorithm 1: PSA

Given: $\eta^{(0)}$, α, β, n;
Compute M and m by Equation (12);
repeat
  Compute $\theta^{(t+1)}$ by Equation (6);
  if $(t+1)$ ***mod*** $2n = 0$ **then**
    Compute a_i for all i by Equation (8) and (9);
    Compute $\eta^{(t+1)}$ by Equation (10) and (11);
  else
    $\eta^{(t+1)} = \eta^{(t)}$;
  end
  $t = t + 1$;
until *Convergence* ;

7. Experiments

We evaluate our proposed method by comparing the performance with SMD. We ran experiments on three data sets: CoNLL-2000 chunking task [14], BioNLP/NLPBA-2004 bio-entity recognition task [5], and BioCreative II gene mention task [18]. We randomly reorder the training data as the input sequences many times to simulate the on-line scenario. The performance is measured by the average F-scores for the hold-out set during the training time. The F-score is defined as $F = \frac{2PR}{P+R}$, where P is the precision and R the recall.

We applied PSA, SMD, and L-BFGS training methods in our experiments. PSA was implemented by modifying CRF++ [1]. We downloaded the SMD program [2] developed by Vishwanathan et al. [17], which was also implemented by revising CRF++. At last, we used CRF++ again for L-BFGS. Though L-BFGS is an off-line algorithm, we ran it to obtain the F-scores every time after processing the whole data set to examine the advantage of on-line methods and to estimate the final F-score that on-line methods are supposed to achieve after convergence.

[1] Available under LGPL from the following URL: `http://crfpp.sourceforge.net/`.

[2] Available under LGPL from the following URL: `http://sml.nicta.com.au/code/crfsmd/`.

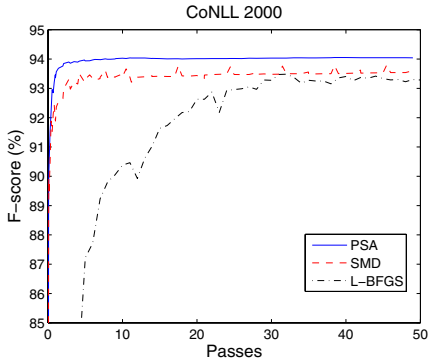

Figure 1. Learning curves of our method, SMD, and L-BFGS on CoNLL-2000 data set.

Here are the parameters for SMD and our method:

- SMD: We used the typical setting described in Vishwanathan et al. [17] for SMD, that is, $\mu = 0.1$, $\lambda = 1.0$, and $\eta^{(0)} = 0.1$. The batch size is fixed to 8 for all data sets.
- PSA: We used $(\alpha, \beta) = (0.9999, 0.99)$, $n = 10$, and $\eta^{(0)} = 0.1$. The batch size is 1 for all data sets.

7.1. CoNLL-2000 Chunking Task

Our first experiment used the data set of the CoNLL-2000 chunking task [14]. The task was to construct a shallow parser for base NP chunking, which labels each word as one of `B` (beginning of chunk), `I` (in a chunk), and `O` (outside a chunk). The training data set contains 8,936 labeled sentences and the test data set contains 2,012 sentences.

We used a CRF model with about 1,000,000 features, which covers the features used by Sha and Pereira [16]. They achieved an F-score of 94.19% with their own extended features.

Figure 1 shows the learning curves of our method, SMD, and L-BFGS for the CoNLL-2000 data set. The learning curves are defined as the function of the progress of F-scores given the number of processed examples, measured by the number of passes through the entire training data sets. We plotted the learning curves in the first 50 passes because both on-line methods converged earlier.

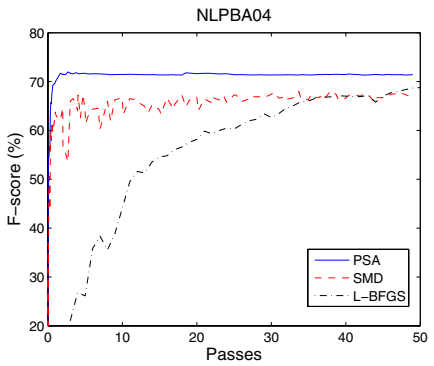

Figure 2. Learning curves of our method, SMD, and L-BFGS on BioNLP/NLPBA-2004 data set.

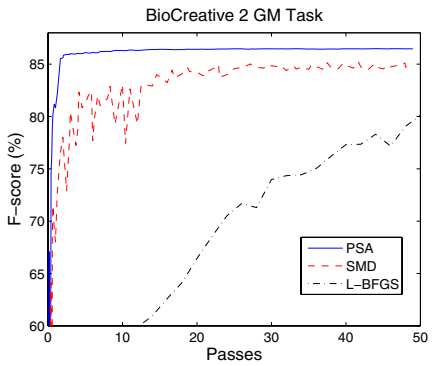

Figure 3. Learning curves of our method, SMD, and L-BFGS on BioCreative II data set.

The F-score of L-BFGS can converge to 94% in about 200 passes, while it arrived at 93.5% in 50 passes. Our method reached an F-score of 93.6% very quickly in about 1.12 passes, and stayed above 93.6% steadily. After 8 passes, the F-score of our method exceeded 94% and finally converged at 94.05%. SMD, reported as the fastest on-line method for CRF, arrived at 93.6% asymptotically in about 7.7 passes in our experiment. SMD once reached 93.8% but finally stayed around 93.6%. The result of SMD is similar to that described in Vishwanathan et al. [17].

From the experiment, we found that our method is about 6.87 times as fast as SMD in terms of passes in achieving the F-score that SMD converged at and our method ended up with a better model than SMD did. The final F-score of our method is almost the same as that of L-BFGS.

7.2. BioNLP/NLPBA-2004 Bio-Entity Recognition Task

Our second experiment used the data set of BioNLP/NLPBA-2004 bio-entity recognition task [5]. The goal of the task is to identify technical terms in the domain of molecular biology and classify them into five categories corresponding to concepts of interest to biologists [5].

We used a CRF model with about 6,000,000 features, most of which follow the features used by Settles [15]. The overall F-score of the system of Settles is 69.8%.

Figure 2 shows the learning curves of our method, SMD, and L-BFGS on the BioNLP/NLPBA-2004 data set. L-BFGS arrived at an F-score of 67% in 40 passes, 70% in 70 passes, and stayed above 70% until convergence. Our method reached an F-score of 67% very quickly in about 0.54 passes, then reached and kept above 70% after processing 1.01 passes. After 1.68 passes, the F-score of our method exceeded 71% and finally converged at 71.4%. SMD arrived at 67% in about 13 passes in our experiment. Then, the F-score of SMD oscillated between 66% and 68% until 50 passes. In Vishwanathan et al. [17], they used different feature sets and achieved a remarkable F-score of more than 85%. However, SMD, with a slightly different setting ($\mu = 0.02$ and $b = 6$), still took about 10 passes to converge.

In this experiment, our method is about 24 times as fast as SMD in terms of passes in achieving the F-score that SMD converged at, and again our method ended up with a better model than SMD did . The quality of our learned model is the same as that of L-BFGS in terms of the F-score.

7.3. BioCreative II Gene Mention Task

Our third experiment used the data set of BioCreative II gene mention task [18]. The goal of the task is to extract all the genes and gene products mentioned in given MEDLINE sentences. In the competition, 15,000 sentences were released as the training data set and 5,000 were left as the test data set for final evaluation.

We employed a CRF model with about 7,300,000 features to label `B`, `I`, and `O` for gene mention extraction. We used most of the features of Kuo et al. [7], who achieved an F-score of 86.83% in the competition by training with L-BFGS, the best result achieved by CRF in that competition.

Figure 3 shows the learning curves of our method, SMD, and L-BFGS for the BioCreative II data set. L-BFGS reached an F-score at 84% in 110 passes, 85% in 156 passes, and stayed above 86% after 200 passes. Our method jumped beyond an F-score of 85% in about 1.66 passes, then reached and kept above 86% after processing 3 passes. After processing the data set for 4 passes, the F-score of our method exceeded 86% and finally converged at 86.46%. SMD arrived at 84% in about 16.67 passes in our experi-

ment. Then, SMD kept the F-score between 84% and 85% until 50 passes.

Our method is about an order of magnitude faster than SMD in terms of passes in achieving the highest F-score of SMD, and finally our method produced a better model than SMD did . The models trained by PSA and L-BFGS achieved competing F-scores.

8. Conclusion

We have proposed PSA, a new step size adaptation method for stochastic gradient descent for on-line large-scale text mining with huge CRF models. We compared the asymptotic behavior of our method with that of SMD, which is currently reported as the fastest method for training CRF on-line. Our method adopts an efficient way to estimate the Jacobian of mappings, and adjust learning rates according to the estimated Jacobian. The adjustment is more efficient than that of SMD. Moreover, our experiments show that training by PSA is 6.87 to 24 times faster than that of SMD in terms of the passes through the training data set, and at the same time, achieves competing F-scores with L-BFGS in all the experiments.

We plan to conduct more large scale experiments on a wide variety of application domains to further evaluate the performance of our method and compare with MIRA and other CRF training algorithms. We also plan to extend our method to other learning problems, such as the kernel based methods, to on-line learning.

Acknowledgements

This work was supported in part by the National Research Program in Genomic Medicine (NRPGM), National Science Council, Taiwan, under Grant No. NSC95-3112-B-001-017 (Advanced Bioinformatics Core), and in part under Grant No. NSC95-2221-E-001-038.

References

[1] K. Crammer and Y. Singer. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3:951–991, January 2003.

[2] J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics*, 43(5):1470–1480, 1972.

[3] H.-S. Huang, B.-H. Yang, and C.-N. Hsu. Triple-jump acceleration for the EM algorithm. In *Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 649–652, 2005.

[4] M. Jamshidian and R. I. Jennrich. Acceleration of the EM algorithm by using quasi-newton methods. *Journal of the Royal Statistical Society, , Series B*, 59(3):569–587, 1997.

[5] J.-D. Kim, T. Ohta, Y. Tsuruoka, Y. Tateisi, and N. Collier. Introduction to the bio-entity recognition task at JNLPBA. In *Proceedings of the Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA-2004)*, pages 70–75, 2004.

[6] T. Kudo. CRF++: Yet another CRF toolkit (`http://crfpp.sourceforge.net/`), 2006.

[7] C.-J. Kuo, Y.-M. Chang, H.-S. Huang, K.-T. Lin, B.-H. Yang, Y.-S. Lin, C.-N. Hsu, and I.-F. Chung. Rich feature set, unification of bidirectional parsing and dictionary filtering for high f-score gene mention tagging. In *Proceedings of the Second BioCreative Challenge Evaluation Workshop*, pages 105–107, 2007.

[8] H. J. Kushner and H. Huang. Asymptotic properties of stochastic approximations with constant coefficients. *SIAM Journal on Control and Optimization*, 19(1):87–105, 1981.

[9] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of 18th International Conference on Machine Learning (ICML'01)*, pages 282–289, 2001.

[10] R. Malouf. A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the Sixth Conference on Natural Language Learning (CoNLL-2002)*, pages 49–55, 2002.

[11] G. J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley Series in Probability and Statistics. Wiley-Interscience, 1997.

[12] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 1999.

[13] R. Salakhutdinov and S. Roweis. Adaptive overrelaxed bound optimization methods. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML'03)*, pages 664–671, 2003.

[14] E. F. T. K. Sang and S. Buchholz. Introduction to the conll-2000 shared task: Chunking. In *Proceedings of Conference on Computational Natural Language Learning (CoNLL-2000)*, pages 127–132, 2000.

[15] B. Settles. Biomedical named entity recognition using conditional random fields and novel feature sets. In *Proceedings of the Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA-2004)*, pages 104–107, 2004.

[16] F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *Proceedings of Human Language Technology, the North American Chapter of the Association for Computational Linguistics (NAACL'03)*, pages 213–220, 2003.

[17] S. Vishwanathan, N. N. Schraudolph, M. W. Schmidt, and K. P. Murphy. Accelerated training of conditional random fields with stochastic gradient methods. In *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, USA, June 2006.

[18] J. Wilbur, L. Smith, and L. Tanabe. Biocreative 2. gene mention task. In *Proceedings of the Second BioCreative Challenge Evaluation Workshop*, pages 7–16, 2007.