Iterative Scaling and Coordinate Descent Methods for Maximum Entropy

Fang-Lan Huang, Cho-Jui Hsieh, Kai-Wei Chang, and Chih-Jen Lin
Department of Computer Science
National Taiwan University
Taipei 106, Taiwan
{d93011,b92085,b92084,cjlin}@csie.ntu.edu.tw

Abstract

Maximum entropy (Maxent) is useful in many areas. Iterative scaling (IS) methods are one of the most popular approaches to solve Maxent. With many variants of IS methods, it is difficult to understand them and see the differences. In this paper, we create a general and unified framework for IS methods. This framework also connects IS and coordinate descent (CD) methods. Besides, we develop a CD method for Maxent. Results show that it is faster than existing iterative scaling methods[1].

1 Introduction

Maximum entropy (Maxent) is widely used in many areas such as natural language processing (NLP) and document classification. Maxent models the conditional probability as:

$$P_{\boldsymbol{w}}(y|x) \equiv S_{\boldsymbol{w}}(x,y)/T_{\boldsymbol{w}}(x), \quad (1)$$

$$S_{\boldsymbol{w}}(x,y) \equiv e^{\sum_t w_t f_t(x,y)}, \; T_{\boldsymbol{w}}(x) \equiv \textstyle\sum_y S_{\boldsymbol{w}}(x,y),$$

where x indicates a context, y is the label of the context, and $\boldsymbol{w} \in R^n$ is the weight vector. A function $f_t(x,y)$ denotes the t-th feature extracted from the context x and the label y.

Given an empirical probability distribution $\tilde{P}(x,y)$ obtained from training samples, Maxent minimizes the following negative log-likelihood:

$$\begin{aligned} &\min_{\boldsymbol{w}} \; -\textstyle\sum_{x,y} \tilde{P}(x,y) \log P_{\boldsymbol{w}}(y|x) \\ &\quad = \textstyle\sum_x \tilde{P}(x) \log T_{\boldsymbol{w}}(x) - \sum_t w_t \tilde{P}(f_t), \end{aligned} \quad (2)$$

where $\tilde{P}(x) = \sum_y \tilde{P}(x,y)$ is the marginal probability of x, and $\tilde{P}(f_t) = \sum_{x,y} \tilde{P}(x,y) f_t(x,y)$ is the expected value of $f_t(x,y)$. To avoid overfitting the training samples, some add a regularization term and solve:

$$\min_{\boldsymbol{w}} L(\boldsymbol{w}) \equiv \sum_x \tilde{P}(x) \log T_{\boldsymbol{w}}(x) - \sum_t w_t \tilde{P}(f_t) + \frac{\sum_t w_t^2}{2\sigma^2}, \quad (3)$$

where σ is a regularization parameter. We focus on (3) instead of (2) because (3) is strictly convex.

Iterative scaling (IS) methods are popular in training Maxent models. They all share the same property of *solving a one-variable sub-problem at a time.* Existing IS methods include generalized iterative scaling (GIS) by Darroch and Ratcliff (1972), improved iterative scaling (IIS) by Della Pietra et al. (1997), and sequential conditional generalized iterative scaling (SCGIS) by Goodman (2002). In optimization, coordinate descent (CD) is a popular method which also *solves a one-variable sub-problem at a time.* With these many IS and CD methods, it is uneasy to see their differences. In Section 2, we propose a unified framework to describe IS and CD methods from an optimization viewpoint. Using this framework, we design a fast CD approach for Maxent in Section 3. In Section 4, we compare the proposed CD method with IS and LBFGS methods. Results show that the CD method is more efficient.

Notation n is the number of features. The total number of nonzeros in samples and the average number of nonzeros per feature are respectively

$$\#\text{nz} \equiv \textstyle\sum_{x,y} \sum_{t: f_t(x,y) \neq 0} 1 \quad \text{and} \quad \bar{l} \equiv \#\text{nz}/n.$$

2 A Framework for IS Methods

2.1 The Framework

The one-variable sub-problem of IS methods is related to the function reduction $L(\boldsymbol{w}+z\boldsymbol{e}_t) - L(\boldsymbol{w})$, where $\boldsymbol{e}_t = [0, \ldots, 0, 1, 0, \ldots, 0]^T$. IS methods differ in how they approximate the function reduction. They can also be categorized according to whether $\boldsymbol{w}$'s components are sequentially or parallely updated. In this section, we create a framework in Figure 1 for these methods.

Sequential update For a sequential-update algorithm, once a one-variable sub-problem is solved, the corresponding element in $\boldsymbol{w}$ is updated. The new $\boldsymbol{w}$ is then used to construct the next sub-problem. The procedure is sketched in

[1] A complete version of this work is at http://www.csie.ntu.edu.tw/~cjlin/papers/maxent_journal.pdf.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 285–288,
Suntec, Singapore, 4 August 2009.

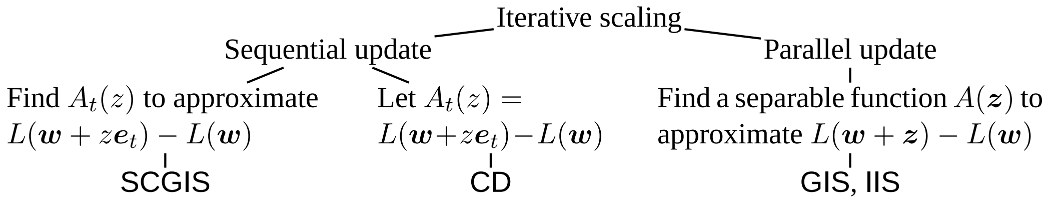

Iterative scaling
Sequential update
Parallel update
Find $A_t(z)$ to approximate $L(\boldsymbol{w}+z\boldsymbol{e}_t)-L(\boldsymbol{w})$
Let $A_t(z)=$ $L(\boldsymbol{w}+z\boldsymbol{e}_t)-L(\boldsymbol{w})$
Find a separable function $A(\boldsymbol{z})$ to approximate $L(\boldsymbol{w}+\boldsymbol{z})-L(\boldsymbol{w})$
SCGIS
CD
GIS, IIS

Figure 1: An illustration of various iterative scaling methods.

Algorithm 1 A sequential-update IS method

```
While $\boldsymbol{w}$ is not optimal
    For $t = 1, \ldots, n$
      1. Find an approximate function $A_t(z)$ satisfying (4).
      2. Approximately $\min_z A_t(z)$ to get $\bar{z}_t$.
      3. $w_t \leftarrow w_t + \bar{z}_t$.
```

Algorithm 2 A parallel-update IS method

```
While $\boldsymbol{w}$ is not optimal
  1. Find approximate functions $A_t(z_t)\ \forall t$ satisfying (7).
  2. For $t = 1, \ldots, n$
       Approximately $\min_{z_t} A_t(z_t)$ to get $\bar{z}_t$.
  3. For $t = 1, \ldots, n$
       $w_t \leftarrow w_t + \bar{z}_t$.
```

Algorithm 1. If the t-th component is selected for update, a sequential IS method solves the following one-variable sub-problem:

$$\min_z \ A_t(z),$$

where $A_t(z)$ bounds the function difference:

$$\begin{aligned} A_t(z) &\geq L(\boldsymbol{w}+z\boldsymbol{e}_t)-L(\boldsymbol{w}) \\ &= \textstyle\sum_x \tilde{P}(x)\log\frac{T_{\boldsymbol{w}+z\boldsymbol{e}_t}(x)}{T_{\boldsymbol{w}}(x)} + Q_t(z) \end{aligned} \qquad (4)$$

$$\text{and} \quad Q_t(z) \equiv \frac{2w_tz+z^2}{2\sigma^2} - z\tilde{P}(f_t). \qquad (5)$$

An approximate function $A_t(z)$ satisfying (4) does not ensure that the function value is strictly decreasing. That is, the new function value $L(\boldsymbol{w}+z\boldsymbol{e}_t)$ may be only the same as $L(\boldsymbol{w})$. Therefore, we can impose an additional condition

$$A_t(0) = 0 \qquad (6)$$

on the approximate function $A_t(z)$. If $A'_t(0) \neq 0$ and assume $\bar{z}_t \equiv \arg\min_z A_t(z)$ exists, with the condition $A_t(0)=0$, we have $A_t(\bar{z}_t)<0$. This inequality and (4) then imply $L(\boldsymbol{w}+\bar{z}_t\boldsymbol{e}_t)<L(\boldsymbol{w})$. If $A'_t(0) = \nabla_t L(\boldsymbol{w}) = 0$, the convexity of $L(\boldsymbol{w})$ implies that we cannot decrease the function value by modifying w_t. Then we should move on to modify other components of $\boldsymbol{w}$.

A CD method can be viewed as a sequential IS method. It solves the following sub-problem:

$$\min_z \ A_t^{\mathsf{CD}}(z) = L(\boldsymbol{w}+z\boldsymbol{e}_t)-L(\boldsymbol{w})$$

without any approximation. Existing IS methods consider approximations as $A_t(z)$ may be simpler for minimization.

Parallel update A parallel IS method simultaneously constructs n independent one-variable sub-problems. After (approximately) solving all of them, the whole vector $\boldsymbol{w}$ is updated. Algorithm 2 gives the procedure. The differentiable function $A(\boldsymbol{z})$, $\boldsymbol{z} \in R^n$, is an approximation of $L(\boldsymbol{w}+\boldsymbol{z})-L(\boldsymbol{w})$ satisfying

$$\begin{aligned} &A(\boldsymbol{z}) \geq L(\boldsymbol{w}+\boldsymbol{z})-L(\boldsymbol{w}), \ \ A(\mathbf{0})=0, \ \text{ and} \\ &A(\boldsymbol{z}) = \textstyle\sum_t A_t(z_t). \end{aligned} \qquad (7)$$

Similar to (4) and (6), the first two conditions ensure that the function value is strictly decreasing. The last condition shows that $A(\boldsymbol{z})$ is separable, so

$$\min_{\boldsymbol{z}} A(\boldsymbol{z}) = \textstyle\sum_t \min_{z_t} A_t(z_t).$$

That is, we can minimize $A_t(z_t), \forall t$ simultaneously, and then update $w_t\ \forall t$ together. A parallel-update method possesses nice implementation properties. However, since it less aggressively updates $\boldsymbol{w}$, it usually converges slower. If $A(\boldsymbol{z})$ satisfies (7), taking $\boldsymbol{z} = z_t\boldsymbol{e}_t$ implies that (4) and (6) hold for any $A_t(z_t)$. A parallel method could thus be transformed to a sequential method using the same approximate function, but not vice versa.

2.2 Existing Iterative Scaling Methods

We introduce GIS, IIS and SCGIS via the proposed framework. GIS and IIS use a parallel update, but SCGIS is sequential. Their approximate functions aim to bound the function reduction

$$L(\boldsymbol{w}+\boldsymbol{z})-L(\boldsymbol{w}) = \textstyle\sum_x \tilde{P}(x)\log\frac{T_{\boldsymbol{w}+\boldsymbol{z}}(x)}{T_{\boldsymbol{w}}(x)} + \sum_t Q_t(z_t), \qquad (8)$$

where $T_{\boldsymbol{w}}(x)$ and $Q_t(z_t)$ are defined in (1) and (5), respectively. Then GIS, IIS and SCGIS use similar inequalities to get approximate functions. They apply $\log\alpha \leq \alpha - 1\ \forall\alpha > 0$ to get

$$(8) \leq \sum_{x,y}\tilde{P}(x)P_{\boldsymbol{w}}(y|x)(e^{\sum_t z_t f_t(x,y)}-1) + \sum_t Q_t(z_t). \qquad (9)$$

GIS defines

$f^{\#} \equiv \max_{x,y} f^{\#}(x,y), \ \ f^{\#}(x,y) \equiv \sum_t f_t(x,y)$, and adds a feature $f_{n+1}(x,y) \equiv f^{\#} - f^{\#}(x,y)$ with $z_{n+1}=0$. Assuming $f_t(x,y) \geq 0, \ \forall t,x,y$, and using Jensen's inequality

$$e^{\sum_{t=1}^{n+1}\frac{f_t(x,y)}{f^{\#}}(z_tf^{\#})} \leq \sum_{t=1}^{n+1}\frac{f_t(x,y)}{f^{\#}}e^{z_tf^{\#}} \ \text{ and}$$

$$e^{\sum_t z_t f_t(x,y)} \leq \sum_t \frac{f_t(x,y)}{f^{\#}}e^{z_tf^{\#}} + \frac{f_{n+1}(x,y)}{f^{\#}}, \qquad (10)$$

we obtain n independent one-variable functions:

$$\begin{aligned} A_t^{\mathsf{GIS}}(z_t) = &\frac{e^{z_tf^{\#}}-1}{f^{\#}}\textstyle\sum_{x,y}\tilde{P}(x)P_{\boldsymbol{w}}(y|x)f_t(x,y) \\ &+ Q_t(z_t). \end{aligned}$$

IIS applies Jensen's inequality

$$e^{\sum_t \frac{f_t(x,y)}{f^\#(x,y)}(z_t f^\#(x,y))} \leq \sum_t \frac{f_t(x,y)}{f^\#(x,y)} e^{z_t f^\#(x,y)}$$

on (9) to get the approximate function

$$A_t^{\mathsf{IIS}}(z_t) = \sum_{x,y} \tilde{P}(x) P_{\boldsymbol{w}}(y|x) f_t(x,y) \frac{e^{z_t f^\#(x,y)}-1}{f^\#(x,y)} + Q_t(z_t).$$

SCGIS is a sequential-update method. It replaces $f^\#$ in GIS with $f_t^\# \equiv \max_{x,y} f_t(x,y)$. Using $z_t \boldsymbol{e}_t$ as $\boldsymbol{z}$ in (8), a derivation similar to (10) gives

$$e^{z_t f_t(x,y)} \leq \frac{f_t(x,y)}{f_t^\#} e^{z_t f_t^\#} + \frac{f_t^\# - f_t(x,y)}{f_t^\#}.$$

The approximate function of SCGIS is

$$A_t^{\mathsf{SCGIS}}(z_t) = \frac{e^{z_t f_t^\#}-1}{f_t^\#} \sum_{x,y} \tilde{P}(x) P_{\boldsymbol{w}}(y|x) f_t(x,y) + Q_t(z_t).$$

We prove the linear convergence of existing IS methods (proof omitted):

Theorem 1 *Assume each sub-problem $A_t^s(z_t)$ is exactly minimized, where s is* IIS, GIS, SCGIS, *or* CD. *The sequence $\{\boldsymbol{w}^k\}$ generated by any of these four methods linearly converges. That is, there is a constant $\mu \in (0,1)$ such that*
$L(\boldsymbol{w}^{k+1}) - L(\boldsymbol{w}^*) \leq (1-\mu)(L(\boldsymbol{w}^k) - L(\boldsymbol{w}^*)), \forall k,$
where $\boldsymbol{w}^*$ is the global optimum of (3).

2.3 Solving one-variable sub-problems

Without the regularization term, by $A_t'(z_t) = 0$, GIS and SCGIS both have a simple closed-form solution of the sub-problem. With the regularization term, the sub-problems no longer have a closed-form solution. We discuss the cost of solving sub-problems by the Newton method, which iteratively updates z_t by

$$z_t \leftarrow z_t - A_t^{s\prime}(z_t) / A_t^{s\prime\prime}(z_t). \tag{11}$$

Here s indicates an IS or a CD method.

Below we check the calculation of $A_t^{s\prime}(z_t)$ as the cost of $A_t^{s\prime\prime}(z_t)$ is similar. We have

$$A_t^{s\prime}(z_t) = \sum_{x,y} \tilde{P}(x) P_{\boldsymbol{w}}(y|x) f_t(x,y) e^{z_t f^s(x,y)} + Q_t'(z_t) \tag{12}$$

where

$$f^s(x,y) \equiv \begin{cases} f^\# \text{ if } s \text{ is GIS}, \\ f_t^\# \text{ if } s \text{ is SCGIS}, \\ f^\#(x,y) \text{ if } s \text{ is IIS}. \end{cases}$$

For CD,

$$A_t^{\mathsf{CD}\prime}(z_t) = Q_t'(z_t) + \sum_{x,y} \tilde{P}(x) P_{\boldsymbol{w}+z_t\boldsymbol{e}_t}(y|x) f_t(x,y). \tag{13}$$

The main cost is on calculating $P_{\boldsymbol{w}}(y|x)\ \forall x,y$, whenever $\boldsymbol{w}$ is updated. Parallel-update approaches calculate $P_{\boldsymbol{w}}(y|x)$ once every n sub-problems, but sequential-update methods evaluates $P_{\boldsymbol{w}}(y|x)$ after every sub-problem. Consider the situation of updating $\boldsymbol{w}$ to $\boldsymbol{w} + z_t\boldsymbol{e}_t$. By (1), obtaining $P_{\boldsymbol{w}+z_t\boldsymbol{e}_t}(y|x)\ \forall x,y$ requires expensive $O(\#\text{nz})$ operations to evaluate $S_{\boldsymbol{w}+z_t\boldsymbol{e}_t}(x,y)$ and $T_{\boldsymbol{w}+z_t\boldsymbol{e}_t}(x)\ \forall x,y$. A trick to trade memory for time is to store all $S_{\boldsymbol{w}}(x,y)$ and $T_{\boldsymbol{w}}(x)$,

$$S_{\boldsymbol{w}+z_t\boldsymbol{e}_t}(x,y) = S_{\boldsymbol{w}}(x,y) e^{z_t f_t(x,y)},$$

$$T_{\boldsymbol{w}+z_t\boldsymbol{e}_t}(x) = T_{\boldsymbol{w}}(x) + \sum_y S_{\boldsymbol{w}}(x,y)(e^{z_t f_t(x,y)} - 1).$$

Since $S_{\boldsymbol{w}+z_t\boldsymbol{e}_t}(x,y) = S_{\boldsymbol{w}}(x,y)$ if $f_t(x,y) = 0$, this procedure reduces the the $O(\#\text{nz})$ operations to $O(\#\text{nz}/n) = O(\bar{l})$. However, it needs extra spaces to store all $S_{\boldsymbol{w}}(x,y)$ and $T_{\boldsymbol{w}}(x)$. This trick for updating $P_{\boldsymbol{w}}(y|x)$ has been used in SCGIS (Goodman, 2002). Thus, the first Newton iteration of all methods discussed here takes $O(\bar{l})$ operations. For each subsequent Newton iteration, CD needs $O(\bar{l})$ as it calculates $P_{\boldsymbol{w}+z_t\boldsymbol{e}_t}(y|x)$ whenever z_t is changed. For GIS and SCGIS, if $\sum_{x,y} \tilde{P}(x) P_{\boldsymbol{w}}(y|x) f_t(x,y)$ is stored at the first Newton iteration, then (12) can be done in $O(1)$ time. For IIS, because $f^\#(x,y)$ of (12) depends on x and y, we cannot store $\sum_{x,y} \tilde{P}(x) P_{\boldsymbol{w}}(y|x) f_t(x,y)$ as in GIS and SCGIS. Hence each Newton direction needs $O(\bar{l})$. We summarize the cost for solving sub-problems in Table 1.

Table 1: Time for minimizing $A_t(z_t)$ by the Newton method

	CD	GIS	SCGIS	IIS
1st Newton direction	$O(\bar{l})$	$O(\bar{l})$	$O(\bar{l})$	$O(\bar{l})$
Each subsequent Newton direction	$O(\bar{l})$	$O(1)$	$O(1)$	$O(\bar{l})$

3 Comparison and a New CD Method

3.1 Comparison of IS/CD methods

From the above discussion, an IS or a CD method falls into a place between two extreme designs:

$A_t(z_t)$ a loose bound $\leftrightarrow$ $A_t(z_t)$ a tight bound
Easy to minimize $A_t(z_t)$ Hard to minimize $A_t(z_t)$

There is a tradeoff between the tightness to bound the function difference and the hardness to solve the sub-problem. To check how IS and CD methods fit into this explanation, we obtain relationships of their approximate functions:

$$\begin{aligned} A_t^{\mathsf{CD}}(z_t) &\leq A_t^{\mathsf{SCGIS}}(z_t) \leq A_t^{\mathsf{GIS}}(z_t), \\ A_t^{\mathsf{CD}}(z_t) &\leq A_t^{\mathsf{IIS}}(z_t) \leq A_t^{\mathsf{GIS}}(z_t) \quad \forall\, z_t. \end{aligned} \tag{14}$$

The derivation is omitted. From (14), CD considers more accurate sub-problems than SCGIS and GIS. However, to solve each sub-problem, from Table 1, CD's each Newton step takes more time. The same situation occurs in comparing IIS and GIS. Therefore, while a tight $A_t(z_t)$ can give faster convergence by handling fewer sub-problems, the total time may not be less due to the higher cost of each sub-problem.

3.2 A Fast CD Method

We develop a CD method which is cheaper in solving each sub-problem but still enjoys fast final convergence. This method is modified from Chang et al. (2008), a CD approach for linear SVM. We approximately minimize $A_t^{\text{CD}}(z)$ by applying only one Newton iteration. The Newton direction at $z = 0$ is now

$$d = -A_t^{\text{CD}\prime}(0)/A_t^{\text{CD}\prime\prime}(0). \tag{15}$$

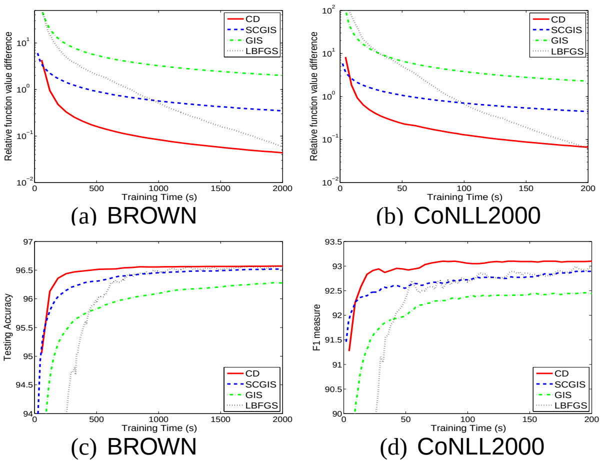
As taking the full Newton direction may not decrease the function value, we need a line search procedure to find $\lambda \geq 0$ such that $z = \lambda d$ satisfies the following sufficient decrease condition:

$$A_t^{\text{CD}}(z) - A_t^{\text{CD}}(0) = A_t^{\text{CD}}(z) \leq \gamma z A_t^{\text{CD}\prime}(0), \tag{16}$$

where γ is a constant in $(0, 1/2)$. A simple way to find λ is by sequentially checking $\lambda = 1, \beta, \beta^2, \ldots,$ where $\beta \in (0, 1)$. The line search procedure is guaranteed to stop (proof omitted). We can further prove that near the optimum two results hold: First, the Newton direction (15) satisfies the sufficient decrease condition (16) with $\lambda = 1$. Then the cost for each sub-problem is $O(\bar{l})$, similar to that for exactly solving sub-problems of GIS or SCGIS. This result is important as otherwise each trial of $z = \lambda d$ expensively costs $O(\bar{l})$ for calculating $A_t^{\text{CD}}(z)$. Second, taking one Newton direction of the tighter $A_t^{\text{CD}}(z_t)$ reduces the function $L(\boldsymbol{w})$ more rapidly than exactly minimizing a loose $A_t(z_t)$ of GIS, IIS or SCGIS. These two results show that the new CD method improves upon the traditional CD by approximately solving sub-problems, while still maintains fast convergence.

4 Experiments

We apply Maxent models to part of speech (POS) tagging for BROWN corpus (`http://www.nltk.org`) and chunking tasks for CoNLL2000 (`http://www.cnts.ua.ac.be/conll2000/chunking`). We randomly split the BROWN corpus to 4/5 training and 1/5 testing. Our implementation is built upon OpenNLP (`http://maxent.sourceforge.net`). We implement CD (the new one in Section 3.2), GIS, SCGIS, and LBFGS for comparisons. We include LBFGS as Malouf (2002) reported that it is better than other approaches including GIS and IIS. We use $\sigma^2 = 10$, and set $\beta = 0.5$ and $\gamma = 0.001$ in (16).



(a) BROWN

(b) CoNLL2000

(c) BROWN

(d) CoNLL2000

Figure 2: First row: time versus the relative function value difference (17). Second row: time versus testing accuracy/F1. Time is in seconds.

We begin at checking time versus the relative difference of the function value to the optimum:

$$L(\boldsymbol{w}) - L(\boldsymbol{w}^*)/L(\boldsymbol{w}^*). \tag{17}$$

Results are in the first row of Figure 2. We check in the second row of Figure 2 about testing accuracy/F1 versus training time. Among the three IS/CD methods compared, the new CD approach is the fastest. SCGIS comes the second, while GIS is the last. This result is consistent with the tightness of their approximation functions; see (14). LBFGS has fast final convergence, but it does not perform well in the beginning.

5 Conclusions

In summary, we create a general framework for explaining IS methods. Based on this framework, we develop a new CD method for Maxent. It is more efficient than existing IS methods.

References

K.-W. Chang, C.-J. Hsieh, and C.-J. Lin. 2008. Coordinate descent method for large-scale L2-loss linear SVM. *JMLR*, 9:1369–1398.

John N. Darroch and Douglas Ratcliff. 1972. Generalized iterative scaling for log-linear models. *Ann. Math. Statist.*, 43(5):1470–1480.

Stephen Della Pietra, Vincent Della Pietra, and John Lafferty. 1997. Inducing features of random fields. *IEEE PAMI*, 19(4):380–393.

Joshua Goodman. 2002. Sequential conditional generalized iterative scaling. In *ACL*, pages 9–16.

Robert Malouf. 2002. A comparison of algorithms for maximum entropy parameter estimation. In *CONLL*.