Newton-CG 法による条件付き確率場のバッチ学習

坪井 祐太　海野 裕也　鹿島 久嗣　岡崎 直観
日本アイ・ビー・エム株式会社　東京大学　東京大学
{yutat,yunno}@jp.ibm.com　kashima@mist.i.u-tokyo.ac.jp　okazaki@is.s.u-tokyo.ac.jp

1　導入

直鎖条件付き確率場 (CRF) は出力ラベル列の予測器として，自然言語処理（NLP）において広く応用されている [4]．CRF は観測 $\boldsymbol{x} \in \boldsymbol{X}$ に対するラベル列 $\boldsymbol{y} \in \boldsymbol{Y}$ の条件付き確率を $P_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x}) = e^{\boldsymbol{\theta}^\top \boldsymbol{\Phi}(\boldsymbol{x},\boldsymbol{y})}/Z$ でモデル化する．ただし，$\boldsymbol{\theta} \in \mathbb{R}^d$ は d 次元のパラメータ・ベクトル，$\boldsymbol{\Phi}(\boldsymbol{x},\boldsymbol{y}) : \boldsymbol{X} \times \boldsymbol{Y} \to \mathbb{R}^d$ は素性関数，分母 $Z = \sum_{\boldsymbol{y} \in Y} e^{\boldsymbol{\theta}^\top \boldsymbol{\Phi}(\boldsymbol{x},\boldsymbol{y})}$ は分配関数である．また，$\boldsymbol{v}^\top$ は $\boldsymbol{v}$ の転置である．訓練データ D が与えられたとき，CRF のバッチ学習は罰則項付き負の対数尤度関数 $f(\boldsymbol{\theta})$ の最小化問題として与えられる．

$$f(\boldsymbol{\theta}) = -\sum_{(\boldsymbol{x},\boldsymbol{y}) \in D} \left[\boldsymbol{\theta}^\top \boldsymbol{\Phi}(\boldsymbol{x},\boldsymbol{y}) - \ln Z\right] + \frac{||\boldsymbol{\theta}||^2}{2\sigma^2} \quad (1)$$

ただし，最終項は過学習を防ぐための平均 0 分散，σ^2 の $\boldsymbol{\theta}$ の正規事前分布である．

非線形関数 $f(\boldsymbol{\theta})$ の最適解は解析的には求められないため，反復法によって最適化する．反復法では，各反復 k において現在のパラメータ $\boldsymbol{\theta}_k$ をある探索方向 $\boldsymbol{s}_k$ に沿って次の値 $\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}_k + \boldsymbol{s}_k$ に更新することを繰り返し，最適解に収束させる．もっとも明らかな探索方向としては勾配方向 $-\boldsymbol{g}_k \equiv -\partial_{\boldsymbol{\theta}} f_k$ がある（最急降下法）．ただし，$f_k \equiv f(\boldsymbol{\theta}_k)$ と略記する．また，別の探索方向としては，Newton 方向 $-\boldsymbol{H}_k^{-1}\boldsymbol{g}_k \in \mathbb{R}^d$ がある．ただし $\boldsymbol{H}_k \equiv \partial_{\boldsymbol{\theta}^2} f_k$ はヘシアン行列である．Newton 方向は f_k の 2 次のテイラー近似: $f(\boldsymbol{\theta}_k + \boldsymbol{s}) \approx f_k + \boldsymbol{g}_k^\top \boldsymbol{s} + \frac{1}{2}\boldsymbol{s}^\top \boldsymbol{H}_k \boldsymbol{s}$ を最小化する更新 $\boldsymbol{s}$ である．2 次近似による最適化手法は非常に少ない反復数で収束することが知られている．

NLP では非常に高次元のパラメータを扱う必要があり，$d \times d$ 行列のヘシアン逆行列 $\boldsymbol{H}^{-1}$ を陽に計算することは現実的でない．そこで，CRF のバッチ学習には記憶制限準 Newton 法 (LBFGS) が広く使われている [8]．LBFGS は，$\boldsymbol{H}^{-1}$ を直近 $m(\ll d)$ のパラメータと勾配の変化履歴（d 次元ベクトル $2m$ 個）で近似し，Newton 方向を求める．

一方，Newton 方向は連立 1 次方程式 $\boldsymbol{H}\boldsymbol{s} = -\boldsymbol{g}$（Newton 方程式）の解でもある．本稿では Newton 方程式を共役勾配 (CG) 法によって解くことで Newton 方向を求める Newton-CG 法 [7] を CRF のバッチ学習に応用することを提案する．なお，本手法でも $\boldsymbol{H}^{-1}$ を陽に持つ必要はない．Newton-CG 法はロジスティック回帰の学習において LBFGS に勝ることが示されており [5]，ロジスティック回帰の一般化である CRF においてもその有効性が期待できる．本研究の貢献は，CRF の a) ヘシアン・ベクトル積 $\boldsymbol{H}\boldsymbol{s}$ の動的計画法による計算手順と，b) 勾配計算の中間結果の再利用による Newton-CG 法の高速化である．基本句チャンキングと固有表現抽出タスクでの実験において，LBFGS より数倍早く収束することが示された．さらに，小・中規模のタスクにおいてはオンライン学習手法よりも早く最良の性能を得られることが確認された．

2　Newton-CG 法

Newton-CG 法は 2 重ループ，a) 現在のパラメータ $\boldsymbol{\theta}_k$ での Newton 方向を CG 法により求める内側のループと，b) Newton 方向に従ってパラメータを $\boldsymbol{\theta}_{k+1}$ に更新する外側のループで構成されている．以降では，内側ループの反復を ℓ，外側ループの反復を k で示す．

内側ループで解く必要のある方程式 $\boldsymbol{H}\boldsymbol{d} = -\boldsymbol{g}$ の解（Newton 方向）は，2 次形式 $q(\boldsymbol{d}) = \frac{1}{2}\boldsymbol{d}^\top \boldsymbol{H}\boldsymbol{d} + \boldsymbol{g}^\top \boldsymbol{d}$ を最小化する解と等しい．ただし，内側ループでは $\boldsymbol{\theta}_k$ は固定であり，勾配ベクトルとヘシアン行列は変わらないため $\boldsymbol{g} \equiv \boldsymbol{g}_k$，$\boldsymbol{H} \equiv \boldsymbol{H}_k$ と略記する．CG 法はすべての $\ell \neq h$ において $\boldsymbol{H}$ に関して直交（$\boldsymbol{d}_\ell \boldsymbol{H} \boldsymbol{d}_h = 0$）する点列 $\boldsymbol{d}_\ell (\ell = 1, 2, \cdots)$ により最悪 d 回の反復数で 1 次方程式を解くことができる．多くの場合 d より少ない反復数で収束することが知られており，$\partial_{\boldsymbol{d}} q$ のノルム $r_\ell = ||\boldsymbol{H}\boldsymbol{d}_\ell + \boldsymbol{g}||$ が停止条件として使われる．また，f のような非線形関数では Newton 方向は最適解近くでのみ早い収束を保障し，初期は厳密に Newton 方向を求める必要がないため，Newton-CG 法では $\boldsymbol{d}_0 = -\boldsymbol{g}_k$ を初期点として $r_\ell \leq \xi_k ||\boldsymbol{g}_k||$ を停止条件とする．ただ

し $0 < \xi_k < 1$ かつ $\lim_{k\to\infty} \xi_k = 0$．つまり，最適解に近づく（勾配が小さくなる）につれて正確に Newton 方向を求める．

外側ループでは，非線形関数 f の最適解に収束することを保障するために，線形探索法や信頼区間法などにより Newton 方向を調整する．4 節の実験では信頼区間法を実装したが，紙面の制限から，Newton-CG 法のアルゴリズム詳細については最適化法の教科書 [7] 等を参照されたい．

Newton-CG 法の計算時間は CG 法が大部分を占めるが，先に述べた適応的な CG 反復の停止条件により初期の反復では勾配方向を使い，反復が進むにつれて厳密に Newton 方向を求めるため計算効率が良い．また，CG 法ではヘシアン行列 $\boldsymbol{H}$ を陽に求める代わりに，$\boldsymbol{H}$ とベクトル $\boldsymbol{d}_\ell$ の積 $\boldsymbol{H}\boldsymbol{d}_\ell \in \mathbb{R}^d$ ベクトルが求められれば q の最小化が行えるため空間効率も良い．次節では，CRF 学習の目的関数（1）のヘシアン・ベクトル積 $\boldsymbol{H}\boldsymbol{d}$ の計算法を提案する．

3　CRF のヘシアン・ベクトル積

直鎖 CRF は κ 次マルコフ性を仮定することで，$O(T|Y|^{\kappa+1})$ 時間でヘシアン・ベクトル積を計算できることを示す．ただし，T はラベル列 $\boldsymbol{y}$ の長さである．なお，CRF の目的関数の勾配 $\boldsymbol{g}$ も同じく $O(T|Y|^{\kappa+1})$ で計算できることが知られている [4]．ヘシアン・ベクトル積は勾配計算時に算出される周辺確率を利用して高速に計算できることも示す．

任意のベクトル $\boldsymbol{d}$ と目的関数のヘシアン行列の積は，事例毎の線形和として次のように分解できる．

$$\boldsymbol{H}\boldsymbol{d} = \left[\sum\nolimits_{(\boldsymbol{x},\boldsymbol{y})\in D} \boldsymbol{H}_{(\boldsymbol{x},\boldsymbol{y})}\boldsymbol{d}\right] + \frac{\boldsymbol{d}}{\sigma^2}$$

また，事例ごとのヘシアン・ベクトル積は次式のとおりである．

$$\boldsymbol{H}_{(\boldsymbol{x},\boldsymbol{y})}\boldsymbol{d} = \sum_{\tilde{\boldsymbol{y}}\in\boldsymbol{Y}} \mathrm{P}_{\boldsymbol{\theta}}(\tilde{\boldsymbol{y}}|\boldsymbol{x})\boldsymbol{\Phi}(\boldsymbol{x},\tilde{\boldsymbol{y}})\boldsymbol{\Psi}_{\boldsymbol{\theta}}(\boldsymbol{x},\tilde{\boldsymbol{y}})^\top\boldsymbol{d} \qquad (2)$$

ただし，$\boldsymbol{\Psi}_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{\Phi}(\boldsymbol{x},\boldsymbol{y}) - \mathbb{E}_{\mathrm{P}_{\boldsymbol{\theta}}}(\boldsymbol{\Phi}(\boldsymbol{x},\boldsymbol{Y}))$．

式 (2) は指数個の要素の集合 $\boldsymbol{Y}$ の和を含むが，CRF の勾配計算と同様に動的計画法で多項式時間で計算可能であることを次に示す．説明のため 1 次のマルコフ性を仮定する．マルコフ性を仮定すると素性関数は $\boldsymbol{\Phi}(\boldsymbol{x},\boldsymbol{y}) = \sum_{t=1}^{T+1}\boldsymbol{\phi}(\boldsymbol{x},y_{t-1},y_t)$ と分解でき，条件付き確率は次のように分解できる．

$$\begin{aligned}\mathrm{P}(\boldsymbol{y}|\boldsymbol{x}) &= \mathrm{P}(y_1|y_2,\boldsymbol{x})\cdots\mathrm{P}(y_{t-2}|y_{t-1},\boldsymbol{x})\\ &\quad \mathrm{P}(y_{t-1},y_t|\boldsymbol{x})\mathrm{P}(y_{t+1}|y_t,\boldsymbol{x})\cdots\mathrm{P}(y_T|y_{T-1},\boldsymbol{x})\end{aligned}$$

ただし，列頭と列末をそれぞれ S と E であらわすとき，$\mathrm{P}(S|y_1,\boldsymbol{x}) = 1$ と $\mathrm{P}(E|y_T,\boldsymbol{x}) = 1$ は省略している．これらの分解により，式（2）は次式で書き直すことができる：

$$\boldsymbol{H}_{(\boldsymbol{x},\boldsymbol{y})}\boldsymbol{d} = \sum_{t=1}^{T}\sum_{i,j\in Y}\boldsymbol{\phi}(\boldsymbol{x},i,j)M(t,i,j).$$

ただし，$M(t,i,j)$ は次式で定義される．

$$\begin{aligned}M(t,i,j) = \mathrm{P}_{\boldsymbol{\theta}}(y_{t-1}{=}i,y_t{=}j|\boldsymbol{x})\,\Big[&\boldsymbol{A}[t{-}1,i]+\\ &\boldsymbol{\phi}(\boldsymbol{x},i,j)^\top\boldsymbol{d} + \boldsymbol{B}[t,j] - \mathbb{E}_{\mathrm{P}_{\boldsymbol{\theta}}}\left(\boldsymbol{\Phi}(\boldsymbol{x},\tilde{\boldsymbol{y}})\right)^\top\boldsymbol{d}\Big].\end{aligned}$$

また，行列 $\boldsymbol{A}[t,j]$ と $\boldsymbol{B}[t,i]$ は次の再帰式で定義される．

$$\boldsymbol{A}[t,j] = \begin{cases} 0 & \text{if } t=0\\ \boldsymbol{\phi}(\boldsymbol{x},S,j)^\top\boldsymbol{d} & \text{else if } t{=}1\\ \sum_{i\in Y}\mathrm{P}_{\boldsymbol{\theta}}(y_{t-1}{=}i|y_t{=}j,\boldsymbol{x}) & \\ \quad\left(\boldsymbol{\phi}(\boldsymbol{x},i,j)^\top\boldsymbol{d} + \boldsymbol{A}[t{-}1,i]\right) & \text{otherwise,}\end{cases}$$

$$\boldsymbol{B}[t,i] = \begin{cases} \boldsymbol{\phi}(\boldsymbol{x},i,E)^\top\boldsymbol{d} & \text{if } t{=}T\\ \sum_{j\in Y}\mathrm{P}_{\boldsymbol{\theta}}(y_{t+1}{=}j|y_t{=}i,\boldsymbol{x}) & \\ \quad\left(\boldsymbol{\phi}(\boldsymbol{x},i,j)^\top\boldsymbol{d} + \boldsymbol{B}[t{+}1,j]\right) & \text{otherwise.}\end{cases}$$

よって，$\boldsymbol{A}$ と $\boldsymbol{B}$ を予め計算しておくことで，式（2）は $O(T|Y|^2)$ で計算可能である．なお，$\boldsymbol{A}$ と $\boldsymbol{B}$ の式に現れる条件付き確率は，周辺確率の割り算

$$\mathrm{P}_{\boldsymbol{\theta}}(y_{t-1}|y_t,\boldsymbol{x}) = \frac{\mathrm{P}_{\boldsymbol{\theta}}(y_{t-1},y_t|\boldsymbol{x})}{\mathrm{P}_{\boldsymbol{\theta}}(y_t|\boldsymbol{x})}$$

で計算できる．$\mathrm{P}_{\boldsymbol{\theta}}(y_{t+1}|y_t,\boldsymbol{x})$ も同様．また，期待値 $\mathbb{E}_{\mathrm{P}_{\boldsymbol{\theta}}}(\boldsymbol{\Phi}(\boldsymbol{x},\boldsymbol{Y}))$ も周辺確率を用いて計算可能である．

重要な点として，この周辺確率は勾配計算時に算出しているため，$\boldsymbol{\theta}$ が固定であれば異なる $\boldsymbol{d}$ でのヘシアン・ベクトル積の計算で共用できる．CRF の周辺確率の計算には四則演算より数十倍遅い指数計算を含んでいる [2]．よって，周辺確率を再利用することにより，四則演算だけで計算できるヘシアン・ベクトル積は勾配計算にわずかな計算を追加するだけで計算できる．2 節で指摘したように Newton-CG 法の内側ループ（CG 法）では $\boldsymbol{\theta}$ は固定であるため，周辺確率を保存して再利用することができ，相性が良い．Newton-CG 法は内側ループの計算量が大部分を占めるため，CRF 全体の学習全体の高速化が期待できる．なお，周辺確率は各事例 $(\boldsymbol{x},\boldsymbol{y})$ について κ 次 CRF では $O(T|Y|^{\kappa+1})$ の保存空間が必要になる．利用可能なメモリ領域に合わせて訓練データ D の部分集合のみの周辺確率をキャッシュすることも可能である．

タスク	訓練	開発	テスト	$\lvert X \rvert$	$\lvert Y \rvert$
チャンキング	8,936	N/A	2,012	338,539	23
固有表現抽出	8,322	1,914	1,516	99,135	9

表 1: データセットの文数・素性数・ラベル数の統計

4 実験

基本句チャンキングと固有表現抽出の 2 つの NLP タスクについて CoNLL2000 [11] および CoNLL2002 [10] のデータセットを使い，提案する Newton-CG 法（以降，NCG）の有効性を検証した．

実験では 1 次の CRF を使用した．チャンキングの素性は CRF++[1] に付属の素性テンプレートと同じ 2 値素性を，固有表現抽出は文献 [1] と同じ 2 値素性を使用した．表 1 にタスク毎の訓練・開発・テスト文数および素性数・ラベル数を示す．また，式（1）のハイパー・パラメータ σ は訓練セットの分割または開発セットを用いて選択した．

比較対象として，バッチ学習には LBFGS を，オンライン学習には確率的勾配法 (SGD) を実装した．SGD の更新式は $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta_k \boldsymbol{g}_k$ である．ただし，$\eta_k > 0$ は学習率であり，勾配 $\boldsymbol{g}_k$ はランダムに選択した 1 事例 $(\boldsymbol{x}, \boldsymbol{y}) \in D$ を用いて次のように定義する．

$$\boldsymbol{g}_k = -\boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{y}) + \mathbb{E}_{\mathrm{P}_{\boldsymbol{\theta}_k}}(\boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{y})) + \frac{\boldsymbol{\theta}_k}{n\sigma^2}$$

なお，$n \equiv |D|$ は全訓練文数である．学習率には k の逆数で減衰する式 $\eta_k = \frac{\eta_0}{1+k/n}$（以降，SGD($1/k$)）[3] と，指数的に減衰する式 $\eta_k = \eta_0 \alpha^{k/n}$（以降，SGD($\alpha^k$)）[12] の 2 種を用いた．ただし，初期学習率 η_0 は最初に 500 文で学習し 500 文で f を評価して選択した．また，文献 [12] より $\alpha = 0.85$ を使用した．さらに，疎な更新を効率的に行うためスカラー u とベクトル $\boldsymbol{v}$ で $\boldsymbol{\theta} = u\boldsymbol{v}$ を実装した [9]．

実装はすべて Java™ で行い，Java HotSpot™ 64-Bit Server VM (1.6.0_18) 上で実行した．CPU は Intel® Core™2 Quad 2.4 GHz，メモリは 8 GB である．

図 1 に，チャンキング・タスクにおける時間毎（横軸）の目的関数の最小値 $f_{\min}$ との差（縦軸）を示す．なお，固有表現抽出タスクの傾向は同じであるため省略する．図中の NCG の c の値は 3 節で示した周辺確率を保存した文数を示す（c=ALL は全文の周辺確率を保存した結果）．また，LBFGS の m はヘシアン逆行列を近似するために差分ベクトルを保存した履歴の長さであり，SGD は $k = 200n$ までの結果を示す．

[1] http://crfpp.sourceforge.net/

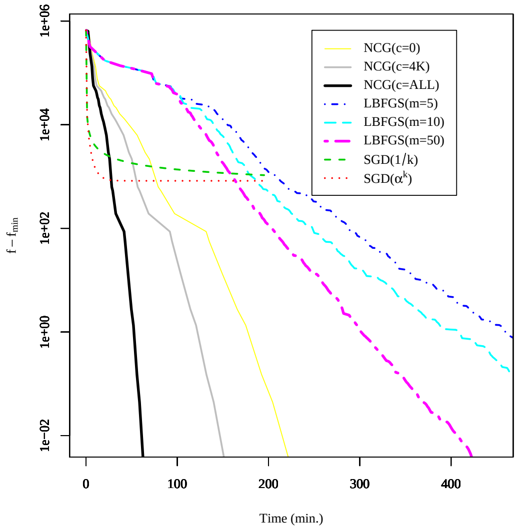


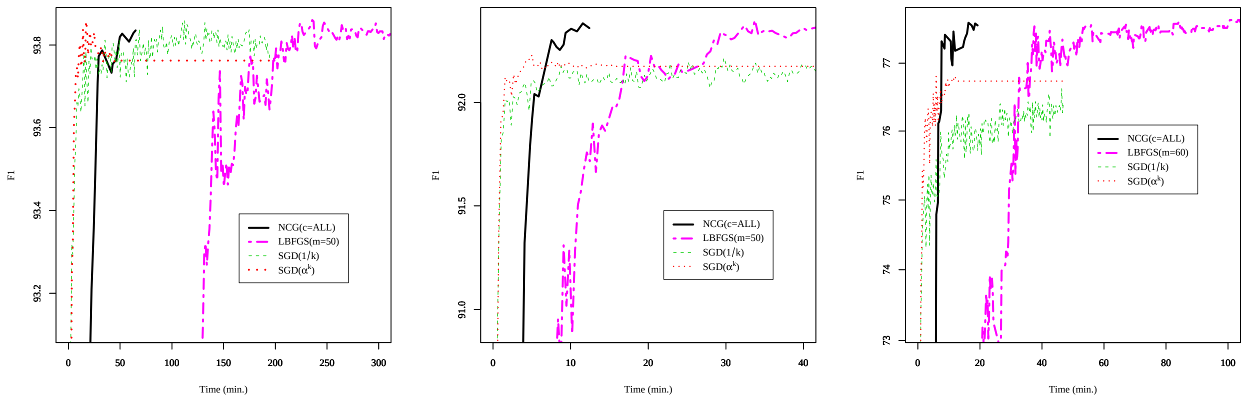
図 1: 目的関数の最小値との誤差 (対数軸) と訓練時間

NCG も LBFGS も c および m を増やすことで収束が早くなっており，空間使用量と計算時間のトレードオフの関係があることがわかる．しかし，NCG(c=ALL) は LBFGS($m = 50$) の 6 倍以上早く収束していることがわかる．なお，搭載メモリの制約から LBFGS の履歴は $m = 50$ 以上増やすことができなかった．この差は，LBFGS は Newton 方向を近似するための情報量が固定であるのに対して，NCG は適応的に Newton 方向の正確さを上げていくことで効率よく最適化できていることに拠ると考えられる．また，SGD は初期に大きく目的関数を減少させるが，後半は減少が緩やかになる（SGD($1/k$)）または停止（SGD(α^k)）した．

次に，図 2 にテスト文での F1 値（縦軸）の時間経過を示す．なお，小規模データでの性能を調べるため，チャンキングタスクのデータを 1/4 にした結果を図 2(b) に示している．また，バッチ学習の NCG と LBFGS は最も速い *NCG*(c=ALL) および *LBFGS*($m = 50$)（チャンキング），*LBFGS*($m = 60$)（固有表現抽出）の結果のみを示す．

図 2 よりすべての場合において，NCG が安定して早く高い F1 値を達成できていることがわかる．LBFGS はすべての場合において最高の F1 値を達成するために NCG の数倍の時間がかかっている．また，SGD は学習の早い段階で高い性能を示すが，必ずしも最高の F1 値を達成できていない．なお，チャンキング全データ（図 2(a)）において SGD(α^k) は NCG より早く最高 F1 値を達成しているが，学習の後半では低い F1 値に収束してしまっている．その他の結果でも SGD(α^k) は早い段階で学習が停まっており，学習率の減衰が早すぎることがわかる．

以上の実験結果より，本実験で使用した規模の学習



(a) 基本句チャンキング　(b) 基本句チャンキング（サイズ 1/4）　(c) 固有表現抽出

図 2: テストデータでの F1 値と訓練時間

タスクでは提案法が最も安定して高速であることが示された．ただし，紙面の都合で省略するが，訓練データ約 4 万文の Penn Treebank の品詞タグ付けタスク [6] では提案法より SGD が最高性能を早く達成できた．しかし，多くの NLP のタスクでは大規模なコーパスを用意することは容易ではなく，提案法は中規模までの NLP タスクの学習に向いているといえる．

5　結論

本研究では，CRF のバッチ学習の高速化により，バッチ学習が有効となる学習データ規模を引き上げることに成功した．小・中規模タスクでは，更新率の調整や停止条件などに課題のあるオンライン学習手法よりも高速で安定した手法であることを実験により示した．今後は，スパース学習やオンライン学習への拡張などを検討したい．

参考文献

[1] Yasemin Altun, Mark Johnson, and Thomas Hofmann. Investigating loss functions and optimization methods for discriminative learning of label sequences. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 145–152, 2003.

[2] Minmin Chen, Yixin Chen, and Michael R. Brent. CRF-OPT: An efficient high-quality conditional random field solver. In *Proceedings of 23rd AAAI Conference on Artificial Intelligence*, pp. 1018–1023, 2008.

[3] Michael Collins, Amir Globerson, Terry Koo, Xavier Carreras, and Peter L. Bartlett. Exponentiated gradient algorithms for conditional random fields and max-margin markov networks. *Journal of Machine Learning Research*, Vol. 9, pp. 1775–1822, 2008.

[4] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, pp. 282–289, 2001.

[5] Chih-Jen Lin, Ruby C. Weng, and S. Sathiya Keerthi. Trust region Newton method for large-scale logistic regression. *Journal of Machine Learning Research*, Vol. 9, pp. 627–650, 2008.

[6] Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, Vol. 19, No. 2, pp. 313–330, 1993.

[7] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer-Verlag, second edition, 2006.

[8] Fei Sha and Fernando Pereira. Shallow parsing with conditional random fields. In *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pp. 134–141, 2003.

[9] Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. Pegasos: Primal Estimated sub-GrAdient SOlver for svm. In *Proceedings of the 24th International Conference on Machine Learning*, pp. 807–814, 2007.

[10] Erik F. Tjong Kim Sang. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2002*, pp. 155–158, 2002.

[11] Erik F. Tjong Kim Sang and Sabine Buchholz. Introduction to the CoNLL-2000 shared task: Chunking. In *Proceedings of CoNLL-2000*, pp. 127–132, 2000.

[12] Yoshimasa Tsuruoka, Jun'ichi Tsujii, and Sophia Ananiadou. Stochastic gradient descent training for L1-regularized log-linear models with cumulative penalty. In *Proceedings of ACL-IJCNLP*, pp. 477–485, 2009.