A Fast Accurate Two-stage Training Algorithm for L1-regularized CRFs with Heuristic Line Search Strategy

Jinlong Zhou
Fudan University
Shanghai, China
abc9703@gmail.com

Xipeng Qiu
Fudan University
Shanghai, China
xpqiu@fudan.edu.cn

Xuanjing Huang
Fudan University
Shanghai, China
xjhuang@fudan.edu.cn

Abstract

Sparse learning framework, which is very popular in the field of nature language processing recently due to the advantages of efficiency and generalizability, can be applied to Conditional Random Fields (CRFs) with L1 regularization method. Stochastic gradient descent (SGD) method has been used in training L1-regularized CRFs, because it often requires much less training time than the batch training algorithm like quasi-Newton method in practice. Nevertheless, SGD method sometimes fails to converge to the optimum, and it can be very sensitive to the learning rate parameter settings. We present a two-stage training algorithm which guarantees the convergence, and use heuristic line search strategy to make the first stage of SGD training process more robust and stable. Experimental evaluations on Chinese word segmentation and name entity recognition tasks demonstrate that our method can produce more accurate and compact model with less training time for L1 regularization.

1 Introduction

Conditional Random Fields (CRFs) (Lafferty et al., 2001; Sutton and McCallum, 2006) are one of the most widely-used machine learning approach in the field of nature language processing, for their ability to handle large feature sets and structural dependency between output labels. The applications of CRFs cover a wide range of tasks such as part-of-speech (POS) tagging (Lafferty et al., 2001), semantic role labeling (Toutannova et al., 2005) and syntactic parsing (Finkel et al., 2008). CRFs outperform other models like Maximum Entropy Markov models (McCallum et al., 2000), because they overcome the problem of "label bias". Moreover, CRFs can output the probabilistic of labeling result for further use as pipeline or reranking.

For all types of CRFs, the maximum-likelihood method can be applied for parameter estimation, which means training the model is done by maximizing the log-likelihood on the training data. To avoid overfitting the likelihood is often penalized with the regularization term. There were two common regularization methods named L1 and L2 regularization. L1 regularization, also called Laplace prior, penalizes the weight vector with its L1-norm. L2 regularization, also called Gaussian prior, uses L2-form. Based on the work of Gao et al. (2007), there is no significant difference between these two regularization methods in terms of accuracy. But L1 regularization has a major advantage that L1-regularized training can produce models, of which the feature weights can be very sparse, then the size of the model will be much smaller than that produced by L2 regularization. Compact models are more interpretable, generalizable and manageable, require less resources like memory and storage. It is very meaningful especially for the rapid development of mobile application nowadays, which suffer the scarcity of resources. In many NLP tasks, the feature sets can reach the magnitude of several million.

Besides, L1 regularization method can implicitly perform the feature selection, and provide the result for further process such as iterative approach (Vail et al., 2007; Peng and McCallum, 2004). This task requires that we need to train the model as accurate as possible, as to converge to the optimum. The feature selection can be regarded as reliable and unbiased after such a process.

Quasi-Newtion method was successfully and efficiently used in L1-regularized model by Andrew and Gao (2007). They presented an algorithm called Orthant-Wise Limited-memory Quasi-Newton (OWL-QN), which is based on L-

BFGS algorithm (Liu and Nocedal, 1989) and achieve better convergence than the method introduced by Kazama and Tsujii (2003).

Stochastic gradient descent (SGD) methods are another kind of L1-regularized training methods. It is a very attractive framework for it often requires much less training time than the batch training algorithm in practice. Tsuruoka et al. (2009) presented a variant of SGD that can efficiently produce compact models with L1 regularization. The main idea is to keep track of the total penalty and the penalty each weight has applied, so that the penalization smooth away the noisy gradient.

Although SGD method with cumulative penalty is very efficient, it sometimes fails to converge to the optimum, because the training process is usually terminated at a certain number of iterations without explicit stop criteria as in quasi-Newton method. Another problem is that the training result of SGD method is very sensitive to the parameter settings of learning rate, therefore we have to tune the values of parameters for different tasks, which is not efficient in practice.

In this paper, we present a two-stage L1-regularized training algorithm to solve these two problems. In the first stage, we use the SGD method to get a relative good solution quickly. In the second stage, we use the OWL-QN method to improve the model which has been dealt with the SGD method. By this means we can fast get the accurate model. The learning rate scheduling in the first stage is done by heuristic line search, which makes the process more robust and stable.

Our experiments are conducted on two tasks, Chinese word segmentation and name entity recognition. We show that our method can produce more accurate and compact model with less training time for L1 regularization. We also verify that the result of SGD training method will be more robust when using the heuristic line search strategy.

The rest of the paper is organized as follows. Section 2 introduces the basics of CRFs. Section 3 describe the two-stage algorithm for L1-regularized models. Experimental results are shown in Section 4. We conclude the work in Section 5.

2 Conditional Random Fields

In this section, we briefly describe the basics of conditional random fields (CRFs) (Lafferty et al., 2001; Sutton and McCallum, 2006) and introduce the definition of some concepts and parameters.

2.1 Linear-chain CRFs

CRFs defines the conditional probabilistic distribution over possible output sequences **y** for observation **x** as following:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left\{\sum_{k=1}^{K} \lambda_k F_k(\mathbf{x},\mathbf{y})\right\}, \qquad (1)$$

where $F_k(\mathbf{x},\mathbf{y})$is equal to $\sum_{t=1}^{T} f_k(\mathbf{x}, y_{t-1}, y_t, t)$. $\{f_k\}$ is a set of feature function and λ_k is the weight of the feature, and $Z(\mathbf{x})$ is the normalization factor defined by

$$Z(\mathbf{x}) = \sum_{\mathbf{y}} \exp\left\{\sum_{k=1}^{K} \lambda_k F_k(\mathbf{x},\mathbf{y})\right\}. \qquad (2)$$

The feature function can be divided into unigram features and bigram features, here we simply rewrite $f_k(\mathbf{x}, y_t, t)$ as $f_k(\mathbf{x}, y_{t-1}, y_t, t)$ for convenient.

2.2 Training

The maximum-likelihood method is a commonly used way applied for parameter estimation, which means we train the model by minimize the negated conditional log-likelihood $L(\lambda)$ on the training data:

$$\begin{aligned} L(\lambda) &= -\sum_{(\mathbf{x},\mathbf{y})} \log p(\mathbf{x},\mathbf{y}) \qquad (3) \\ &= \sum_{(\mathbf{x},\mathbf{y})} \left\{\log Z(\mathbf{x}) - \sum_{k=1}^{K} \lambda_k F_k(\mathbf{x},\mathbf{y})\right\}. \end{aligned}$$

To avoid overfitting, the likelihood is often penalized with the regularization term, which we will talk about in the later sections.

The partial derivative of $L(\lambda)$ by the feature weights λ_k are given by

$$\begin{aligned} \frac{\partial}{\partial \lambda_k} L &= \sum_{(\mathbf{x},\mathbf{y})} \sum_{t=1}^{T} E_{p(\mathbf{y}|\mathbf{x})} f_k(\mathbf{x}, y_{t-1}, y_t, t) \\ &\quad - \sum_{(\mathbf{x},\mathbf{y})} \sum_{t=1}^{T} f_k(\mathbf{x}, y_{t-1}, y_t, t) \qquad (4) \end{aligned}$$

where $E_{p(\mathbf{y}|\mathbf{x})}$ denotes the conditional expectation under the model distribution:

$$\begin{aligned} &E_{p(\mathbf{y}|\mathbf{x})} f_k(\mathbf{x}, y_{t-1}, y_t, t) = \\ &\sum_{(y',y)} f_k(\mathbf{x}, y', y, t) P(y_{t-1} = y', y_t = y|\mathbf{x}) \qquad (5) \end{aligned}$$

Computing the conditional expectation directly is impractical for the large number of possible tag sequences, which is exponential in the length of the observation. Thus, a dynamic programming approach known as the Forward-Backward algorithm originally described for Hidden Markov models (Rabiner, 1989), is applied in a slightly modified form. For the forward recursions, we have

$$\alpha_0(\perp) = 1$$
$$\alpha_{t+1}(y) = \sum_{y'} \alpha_t(y') \exp\left\{\sum_{k=1}^{K} \lambda_k f_k(\mathbf{x}, y', y, t)\right\}$$

and for the backward recursion, we have

$$\beta_{T+1}(\top) = 1$$
$$\beta_t(y') = \sum_{y} \beta_{t+1}(y) \exp\left\{\sum_{k=1}^{K} \lambda_k f_k(\mathbf{x}, y', y, t)\right\}$$

for $0 \leq t \leq T$ and $y \in Y$, where $\perp$ and $\top$ are defined as special states for the begin and end of the sequence. Then the normalization factor is computed by

$$Z(\mathbf{x}) = \beta_0(\perp), \tag{6}$$

and the conditional probabilities $P(y_{t-1} = y', y_t = y|\mathbf{x})$ are given by

$$\alpha_t(y') \exp\left\{\sum_{k=1}^{K} \lambda_k f_k(\mathbf{x}, y', y, t)\right\} \beta_{t+1}(y)/Z(\mathbf{x})$$

3 L1 Regularization in CRFs

3.1 Regularization

The logarithmic loss function $L(\lambda)$ defined by (3) is usually penalized with an additional regularization term, which prevents the model from overfitting the training data. There are two common regularization methods named L1 and L2 regularization, in the case of L1 regularization, the term is defined as:

$$R(\lambda) = C \sum_{k} |\lambda_k|, \tag{7}$$

where C is the regularization parameter that controls the trade-off between fitting exactly the observations and the L1-norm of the weight vector. This value is usually tuned by cross-validation or using the heldout data.

Now we can redefine the objective loss function as

$$L(\lambda) + R(\lambda). \tag{8}$$

3.2 Orthant-Wise Limited-memory Quasi-Newton

It is not easy to use some common numerical optimization strategies such as limited memory BFGS (Liu and Nocedal, 1989) directly with L1 regularization, because the regularization term is not differentiable when the weight is zero.

A very efficient strategy called Orthant-Wise Limited-memory Quasi-Newton (OWL-QN) method is introduced in (Andrew and Gao, 2007). This algorithm is motivated by the observation that the L1 regularization term is differentiable when restricted to a set of points in which each coordinate never changes sign (called its "orthant"). Furthermore it is a linear function of its argument, which means the second-order behavior of the regularized objective function on a given orthant is determined by the log-likelihood component alone. Only a few steps of the standard L-BFGS algorithm have been changed in the OWL-QN method, and these differences are listed below:

1. The "pseudo-gradient" is used in place of the gradient.
2. The resulting search direction is constrained to match the sign pattern of the negated pseudo-gradient.
3. Each parameter is projected back onto the initial orthant of the previous value during the line search.

Andrew and Gao (2007) proved that OWL-QN method is guaranteed to converge to a globally optimal result.

3.3 Stochastic Gradient Descent

Stochastic gradient approaches use a small batch of the observations to get a crude approximation of the gradient of the objective function given by (3). The small batch size makes us possible to update the parameters more frequently than the origin gradient descent and speed up the convergence. Only considering the log-likelihood term, the updates have the following form

$$\hat{\lambda}_k^j = \lambda_k^j + \eta_j \left.\frac{\partial L(\lambda)}{\partial \lambda_k}\right|_{\lambda=\lambda^j} \tag{9}$$

where j is the iteration counter and η_j is the learning rate. It should be noted that the partial derivative we presented here is not the true gradient but the crude approximation from small randomly-selected subset of the training samples. In (Tsuruoka et al., 2009), a variant of SGD that can efficiently train L1-regularized CRFs was presented. The main idea can be concluded as follows:

1. Only update the weights of the features that are used in the current observation, called "lazy update".
2. "Clip" the parameter value when it crosses zero.
3. Keep track of the cumulated penalty that the weights of features should been received if the fluctuationless gradient were used, and use this value to the update.

Let z_j be the total L1-penalty that each weight should been received, it is simply accumulated as:

$$z_j = \frac{C}{N}\sum_{t=1}^{j}\eta_t. \tag{10}$$

Then the process of regularization can be formalized as follows

$$\lambda_k^{j+1} = \begin{cases} \max(0, \hat{\lambda}_k^j - (z_j + q_k^{j-1}) & \hat{\lambda}_k^j > 0 \\ \min(0, \hat{\lambda}_k^j + (z_j - q_k^{j-1}) & \hat{\lambda}_k^j < 0 \end{cases}$$

where q_k^j is the total L1-penalty that λ_k has actually received:

$$q_k^j = \sum_{t=1}^{j}(\lambda_k^{t+1} - \hat{\lambda}_k^t). \tag{11}$$

Tsuruoka et al. (2009) demonstrated that this algorithm can be much more quickly than the OWL-QN method and yield a comparable performance, while the value of objective function and the number of active features are not as good as OWL-QN. The reason is that we usually terminated the training process at a certain number of iterations, because there are no explicit stop criteria for SGD.

Another issue is that the scheduling of learning rates can be very tricky. Tsuruoka et al. (2009) suggest that exponential decay is a good choice in practice compared with the method used in (Collins et al., 2008). This kind of scheduling of learning rates have the following form:

$$\eta_j = \eta_0\alpha^{j/N}, \tag{12}$$

where η_0 and α are both constant. We name η_0 the initiation learning rate parameter and α the descent learning rate parameter. These learning rate parameters have a great influence on the result of SGD training, and they need to be tuned for different tasks, which is not very efficient in practice.

3.4 Two-stage L1-regularized Training

Based on what we have mentioned above, we know that the SGD method sometimes fails to converge to the global optimal solution, for it does not have the explicit stop criteria as in the quasi-Newton method. Although the scheduling of learning rate found in (Collins et al., 2008):

$$\eta_j = \frac{\eta_0}{1 + j/N} \tag{13}$$

guarantees ultimate convergence theoretically. Its actual convergence speed is poor in practice (Darken and Moody,1990). We have to take quite a number of iterations if we need the result close enough to the best solution. This contradicts to the main motivation we use SGD method for parameter estimation that can speed up the training process.

On the other hand, based on the work of Andrew and Gao (2007), we know that OWL-QN method guarantees the convergence. And we can test the relative change in the objective function value averaged over the several previous iterations for stop criteria.

Tsuruoka et al. (2009) demonstrated that SGD method converges much faster than OWL-QN method especially in the first few iterations. This fact motivates us to use a two-stage training strategy. In the first stage, we use the SGD method to quickly get a relative good solution. In the second stage, we use the OWL-QN method to improve the model which has been dealt with the SGD method.

This method can be also driven from an alternative view. In the theory of convex optimization (Boyd and Vandenberghe, 2004), the asymptotic convergence rate of Newton's method is quadratic if we start at a point close enough to the global optimum.[1] In fact, the iterations in Newton's method can fall into two stages. The second stage, which occurs once the searching point is quite close to the optimum solution, is called "quadratically convergent stage". The first stage is usually referred as the "damped Newton phase", because the algorithm may choose a step size that is different from the exact Newton step to satisfy the backtracking

[1]Quasi-Newton method shares many properties with Newton's method, though its convergence rate is generally superlinear but not quadratic.

condition.[2] The quadratically convergent stage is also called the "pure Newton phase" since the full Newton step is always chosen in these iterations . This fact demonstrates that if we can find a solution close to the global optimum, it will not only increase the average convergence rates, but also reduce the time consuming needed for backtracking line search. This is what we achieved by the first stage of SGD training method.

3.5 Heuristic Line Search

Another problem of SGD method is the troublesome learning rate parameters tuning, and these parameters have a significant influence on the result of SGD training. No matter which way to set the learning rate, if it is fixed without taking the actual effect of the current training sample update into consideration, it may be too large or too small for some situation. In order to get a more robust and stable method for learning rate scheduling, we present a heuristic line search strategy inspired by the implementation of CRFsgd (Bottou, 2007) for learning rate calibration.

For the purpose of convenience, we define the objective function for a single sample as

$$l(\lambda, \mathbf{x}) = -\log p(\mathbf{x}, \mathbf{y}) + \frac{C}{N} \sum_{\lambda_k \in \mathbf{X}} |\lambda_k|. \quad (14)$$

Notice here we only use these active features in the current sample as the L1 regularization term, for we only update these associate parameters in a lazy fashion.

Now we try to find the learning rate that decrease the value of this objective function as much as possible without consuming too much search time. We simply use a heuristic line search strategy as follows: (1) We use the learning rate calculated by Eq.12 as initiation and get the initial value of Eq.14. (2) Then we go on to increase the learning rate until the maximum number of trials for search is reached or the value of Eq.14 is worse than the initial value, we just decrease the learning rate from the initiation if the latter situation happens. (3) At last we just use the learning rate that yields the best result of Eq.14 during the search. For the calculation of Eq.14 only needs the weights of the features that are used in the current sample, so it will still be very efficient. The whole algorithm in pseudo-code was showed in Algorithm 1.

[2]To ensure the objective function decrease a certain value and guarantee the convergence.

Algorithm 1 SGD heuristic line search

```
 1. for $k$ = 0 to MaxIterations
 2.    Select sample $j$
 3.    $bestr \leftarrow$ LearningRate($k$)
 4.    UpdateWeights ($j$,$bestr$)

 5. procedure LearningRate($k$)
 6.    $r(0) \leftarrow$ initialed by Eq.12
 7.    $init\_obj \leftarrow$ initialed by Eq.14
 8.    for $i$ = 0 to MaxTrialTime
 9.       UpdateWeights($j$,$r(i)$)
10.       $obj(i) \leftarrow$ Eq.14
11.       Recover weights before update
12.       if $obj(i)$ is worse than $init\_obj$ then
13.          label $flag$
14.       if $flag$ status is changed then
15.          $r(i+1) \leftarrow r(0) * decay$
16.       else
17.          if $flag$ is not set then
18.             $r(i+1) \leftarrow r(i)/decay$
19.          else
20.             $r(i+1) \leftarrow r(i) * decay$
21.    $bestr = argmin_{r(i)} obj(i)$
22.    return $bestr$.
```

It should be noted that we need not set a large number for maximum trial time, because it will generally take a lot of search time and may not yield a good result but arrives at the local optimum, for we only optimize the objective value of a single training sample. Here we set the value to 3 empirically. The changing rate for line search can be any positive number that smaller than 1, it is set to 0.5 as mostly accepted.

4 Experiments

We evaluate the effectiveness and performance of our training algorithm using two NLP tasks that includes Chinese words segmentation and name entity recognition, which are very typical problems in the field of NLP.

To show the improvement of our algorithm, we compare it with the OWL-QN algorithm and SGD algorithm on the same data sets. For the purpose of run-times comparison, we implemented all the algorithm in a quite similar way, especially in feature extraction and gradient computation. For example, we compute the forward/backward scores in logarithm domain instead of scaling

Table 1: Feature templates for Chinese word segmentation task.

(1) $c_{i-1}y_i$, c_iy_i, $c_{i+1}y_i$
(2) $c_{i-1}c_iy_i$, $c_ic_{i+1}y_i$, $c_{i-1}c_{i+1}y_i$
(3) $y_{i-1}y_i$

method, though the latter method was claimed faster (Lavergne et al., 2010). All experiments were performed on a server with Xeon 2.66GHz.

4.1 Chinese Word Segmentation

The first set of experiments used the Chinese word segmentation corpus from the Second International SIGHAN Bakeoff data sets (Emerson, 2005), provided by Peking University. The training data consists of 19,054 sentences, 1,109,947 Chinese words, 1,826,448 Chinese characters and the testing data consists of 1,944 sentences, 104,372 Chinese words, 172,733 Chinese characters. We separated 1,000 sentences from the training data and use them as the heldout data. The test data was only used for the final accuracy report.

The feature templates we used in this experiment were listed in Table 1, where c_i denotes the i^{th} Chinese character in an instance, y_i denotes the i^{th} label in the instance. Based on the work of Huang and Zhao (2007), it was shown that 6 label representation is a better choice in practice. Compare with the origin 2 label representation or 4 label representation, it can represent richer label information. We did not use any extra knowledge such as Chinese and Arabic numbers.

For OWL-QN method and SGD method, we followed the experiment settings in (Tsuruoka et al., 2009). The meta-parameters for OWL-QN method were the same with the default settings of the optimizer developed by Andrew and Gao (2007), the convergence tolerance was 1e-4; the L-BFGS memory parameter was 10. The regularization parameter C was tuned in the way that it maximized the log-likelihood of the heldout data when using the OWL-QN algorithm. We also used this value as the regularization parameter in the SGD method. The learning rate parameters for SGD were tuned in the way that they maximized the value of the objective function in 30 passes. We first set the initiation learning rate parameter (η_0) by testing 1.0, 0.5, 0.2, and 0.1, then we set the descent learning rate parameter (α) by testing 0.9, 0.85, and 0.8 with the fixed initiation learning rate parameter.

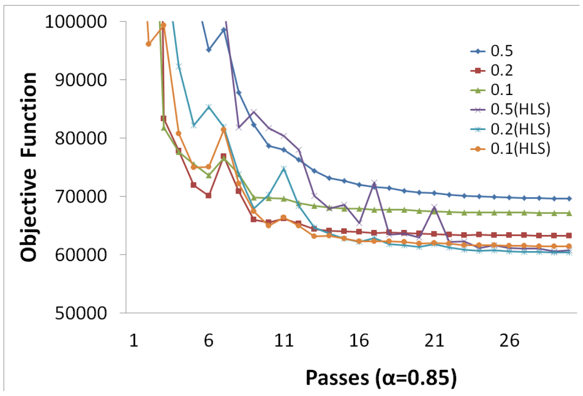

Figure 1: Bakeoff 2005 Chinese word segmentation task: Objective function with fixed α.

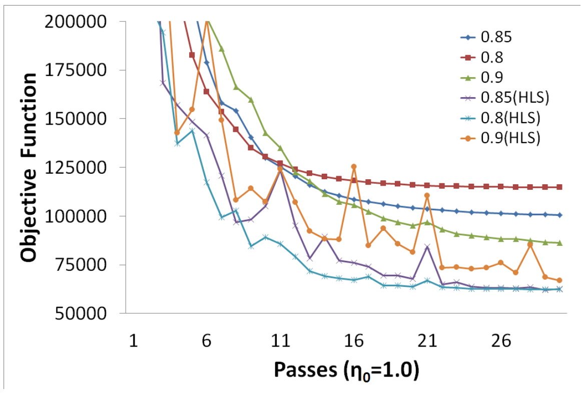

Figure 2: Bakeoff 2005 Chinese word segmentation task: Objective function with fixed η_0.

For our method, we first measured the progress of the SGD algorithm with heuristic line search we presented against the origin SGD method. We use the same parameters settings with the former method including both regularization parameter and learning rate parameters. The number of passes performed over the training data was also set to 30. Then we compare the results of both methods during the training process of the model with the same parameters, and they were shown in Figure 1 and Figure 2.

Figure 1 shows how the value of the objective function changed as the training proceeded with the same descent learning rate parameter ($\alpha = 0.85$), the figure contains six curves representing the results of SGD method with heuristic line search and the origin SGD method with different initiation learning rate parameter settings (η_0). "HLS" stands for the heuristic line search strategy. The results shows SGD method with heuristic line search shows better convergence and more robust

Table 2: Bakeoff 2005 Chinese word segmentation task. Accuracy of the model on the testdata.

	$L + R$	# Features	F score
OWL-QN	56451.8	114,942	94.81
SGD	61398.6	232,585	94.92
Ours	56481.9	117,374	94.78

Table 3: Bakeoff 2005 Chinese word segmentation task. Training time of the model on the testdata.

	Passes	Time
OWL-QN	141	2h59min
SGD	30	58min
Ours	5 + 88	2h06min

result than the origin SGD method when using the same learning rate parameter settings. Figure 2 shows the results with different settings of learning rate parameters (fixed $\eta_0 = 1$), and it demonstrates the same trend as Figure 1.

Then we trained the models with the training data and evaluated the accuracy of the Chinese word segmenter on the test data. The number of passes performed over the training data in SGD was also set to 30. In our method, we set the SGD iteration times to 5. It is worth noting that we didn't spend much time in tuning the value of this parameter. Based on a cursory view of the training process, we found that it converge to a relative "good" result after the first 5 iteration. We used this value throughout all the experiments. Because we would take the OWL-QN method to guarantee the final convergence, and the SGD method with heuristic line search strategy is insensitive to the learning rate parameters, this value would not have a significant influence on the performance.

The results are shown in Table 2 and Table 3. In Table 2, the second column shows the final value of the objective function. The third column shows the number of active features in the final resulting model. The fourth column shows the F score of the Chinese word segment results, which is the harmonic mean of precision P (percentage of output Chinese words that exactly match the golden standard Chinese words) and recall R (percentage of golden standard Chinese words that returned by our system). In Table 3, the second column shows the number of passes performed in the training, in our method, this value includes the the number of passes both in the first stage of SGD and the second stage of OWL-QN process. The third column shows the training time.

Table 4: Feature templates for name entity recognition task.

(1) $c_{i-2}y_i$, $c_{i-1}y_i$, c_iy_i, $c_{i+1}y_i$, $c_{i+2}y_i$
(2) $c_{i-1}c_iy_i$, $c_ic_{i+1}y_i$
(3) $y_{i-1}y_i$

In the terms of accuracy, there was no significant difference between all the models, the origin SGD method yield the slightly better result, probably due to the model has larger features sets. This doesn't contradict to our original purpose, for we have got a substantial improved result in both the final value of the objective function and the number of active features compared with the origin SGD method, and to the same level as OWL-QN method. Notice the origin feature sets are over 6 millon, L1 regularization methods produced the models which are compact indeed. The official best result in the closed test achieved an F score of 95.00, and our result is quite close to that, ranked 4th of 23 official runs.

On the other hand, our method took about 30% less than the OWL-QN method in the training time. Our method only needs 88 passes over the whole training data in the second stage for convergence compared with 141 in the OWL-QN method, which shows a significant improvement in training time consuming, for we have used the first stage of SGD method to get a nearly optimal and stable result beforehand.

4.2 Name Entity Recognition

The second set of experiments used the name entity recognition corpus from the Fourth International SIGHAN Bakeoff data sets (Jin and Chen, 2008), provided by Microsoft Research Asia. The training data consists of 23,182 sentences, 1,089,050 Chinese characters and the testing data consists of 4,636 sentences, 219,197 Chinese characters. We separated 1,000 sentences from the training data and use them as the heldout data. The training data is annotated with the "IOB" tags representing name entities including person, location and organization.

The feature templates we used in this experiment were listed in Table 4. Notice we did not change the label representation made by the origin

Table 5: Fourth SIGHAN Bakeoff name entity recognition task. Training time of the model on the testdata.

	$L+R$	Passes	Time
OWL-QN	11247.1	219	5h26min
SGD	13993.3	30	1h08min
Ours	11245.5	5 + 122	3h10min

Table 6: Fourth SIGHAN Bakeoff name entity recognition task. Accuracy of the model on the testdata.

	# Feat.	LOC	ORG	PER
OWL-QN	34,579	89.94	82.61	90.65
SGD	113,005	89.39	82.75	90.78
Ours	36,709	90.05	82.25	90.49

training data for convenient. Again a richer label representation may yield a better performance.

The other experiment settings are the same with the experiment on Chinese word segmentation. The comparison results are shown in Table 5, Figure 3 and Figure 4. The trend in the results is the same as that of the Chinese word segmentation task. SGD method with heuristic line search strategy produced more stable and robust result than the origin SGD method. Although there will have fluctuations sometimes (in Figure 4), the line search strategy shows the ability to find an appreciate step size in that case. Again our method converged to a much better solution against SGD in both the final value of the objective function and number of active features, and took about 40% less training time than OWL-QN.

The accuracy of the results is shown in Table 6, there was no significant difference between all the models as well. The F score of organization name entity recognition was worse than the results in person and location name entity, for organization name entities in Chinese often have a relative long distance dependency, which is not easy to be captured by our local feature templates in the Chinese character level.

5 Conclusion

We have presented a two-stage algorithm that can efficiently train L1-regularized CRFs. Experiments on two NLP tasks demonstrated that our method is effective and efficient by utilizing both the advantages of SGD and OWL-QN.

In the future, we intend to study how to use the results of the first stage of SGD learning to estimate the Hessian information, which can be provided for the second stage of quasi-Newton method to enhance the effectiveness of training. Borders et al. (2009) looked into this problem in a similar way. It is also worthwhile to investigate whether other adaptive learning rate scheduling algorithms can result in fast training with our method, as in (Vishwanathan et al., 2006; Huang et al., 2007).

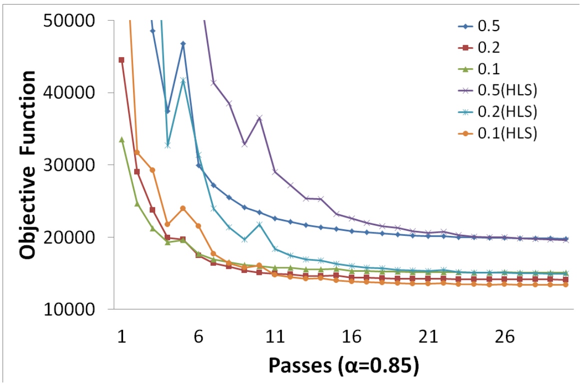

Figure 3: Fourth SIGHAN Bakeoff name entity recognition task: Objective function with fixed α.

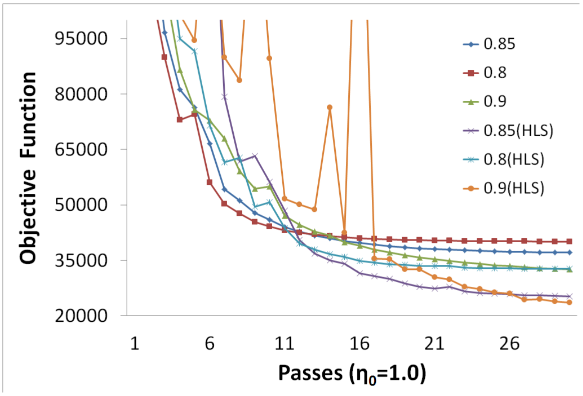

Figure 4: Fourth SIGHAN Bakeoff name entity recognition task: Objective function with fixed η_0.

Acknowledgments

The author wishes to thank the anonymous reviewer for their helpful suggestions. This work was (partially) funded by NSFC (No. 61003091 and No. 61073069) and Shanghai Committee of Science and Technology (No. 10511500703)

References

Galen Andrew and Jianfeng Gao. 2007. Scalable training of l1-regularized log-linear models. In *Proceedings of the International Conference on Machine Learning*, pages 33–40. Corvalis, Oregon, USA.

Antoine Bordes, Léon Bottou, and Patrick Gallinari. 2009. SGD-QN: Careful quasi-Newton stochastic gradient descent. In *The Journal of Machine Learning Research*, 10: 1737–1754.

Léon Bottou. 2007. Stochastic gradient descent (sgd) implementation. http://leon.bottou.org/projects/sgd.

Stephen Boyed and Lieven Vandenberghe. 2004. *Convex Optimiaztion*. Cambridge University Press.

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

Christian Darken and John Moody. 1990. Note on learning rate schedules for stochastic optimization. In *Proceedings of Advances in Neural Information Processing Systems 3*, pages 832–838. Colorado, USA.

Tom Emerson. 2005. The second international Chinese word segmentation bakeoff. In *Proceedings of Fourth SIGHAN Workshop on Chinese Language Processing*. Korea.

Jenny Rose Finkel, Alex Kleeman, and Christopher D.Manning. 2008. Efficient, feature-based, conditional random field parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 959–967. Columbus, Ohio, USA.

Jianfeng Gao, Galen Andrew, Mark Johnson, and Kristina Toutanova. 2007. A comparative study of parameter estimation methods for statistical natural language processing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 824–831. Prague, Czech republic.

Changning Huang and Hai Zhao. 2007. Chinese word segmentation: A decade review. In *Journal of Chinese Information Processing*, 21(3): 8–19.

Han-Shen Huang, Yu-Ming Chang, and Chun-Nan Hsu. 2007. Training conditional random fields by periodic step size adaptation for large-scale text mining. In *Proceedings of the IEEE International Conference on Data Mining*, pages 511–516. Omaha, Nebraska, USA.

Guangjin Jin and Xiao Chen. 2008. The fourth international Chinese language processing bakeoff: Chinese word segmentation, named entity recognition and Chinese pos tagging. In *Proceedings of Sixth SIGHAN Workshop on Chinese Language Processing*. India.

Junichi Kazama and Junichi Tsujii. 2003. Evaluation and extension of maximum entropy models with inequality constraints. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 137–144.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning*, pages 282–289. Prague, Czech republic.

Thomas Lavergne, Olivier,Cappé, and François Yvon. 2010. Practical very large scale CRFs. In *Proceedings the Annual Meeting of the Association for Computational Linguistics*, pages 504–513. Uppsala, Sweden.

Dong C. Liu and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45: 503–528.

Andrew Mccallum, Dayne Freitag, and Fernando Pereira. 2000. Maximum Entropy Markov Models for Information Extraction and Segmentation. In *Proceedings of the International Conference on Machine Learning*, pages 591–598. California, USA.

Naoaki Okazaki. 2007. CRFsuite: A fast implementation of conditional random fields (CRFs). http://www.chokkan.org/software/crfsuite/.

Fuchun Peng, and Andrew McCallum. 2004. Accurate Information Extraction from Research Papers using Conditional Random Fields. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, pages 329–336. Massachusetts, USA.

Lawrence Rabiner. 1989. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In *Proceedings of the IEEE*, 77(2): 257–286.

Charles Sutton and Andrew McCallum. 2006. An introduction to conditional random fields for relational learning. *Introduction to Statistical Relational Learning*. The MIT Press.

Kristina Toutanova, Aria Haghighi, and Christopher Manning. 2005. Joint learning improves semantic role labeling. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 589–596. Michigan, USA.

Yoshimasa Tsuruoka, Junichi Tsujii, and Sophia Ananiadou. 2009. Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty. In *Proceedings the Annual Meeting of the Association for Computational Linguistics*, pages 477–485. Suntec, Singapore.

Douglas Vail, John Lafferty, and Manuela Veloso. 2007. Feature Selection in Conditional Random Fields for Activity Recognition. In *Proceedings of* *the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3379–3384. California, USA.

S. V. N. Vishwanathan, Nicol N. Schraudolph, Mark W. Schmidt, and Kevin P. Murphy. 2006. Accelerated training of conditional random fields with stochastic gradient methods. In *Proceedings of the International Conference on Machine learning*, pages 969–976. Pittsburgh, Pennsylvania, USA.