Bidirectional Sequence Labeling via Dual Decomposition

Zhiguo Wang[1], Chengqing Zong[1] and Nianwen Xue[2]

[1]National Laboratory of Pattern Recognition,
Institute of Automation, Chinese Academy of Sciences, Beijing, China, 100190
[2]Computer Science Department, Brandeis University, Waltham, MA 02452

{zgwang, cqzong}@nlpr.ia.ac.cn xuen@brandeis.edu

Abstract. In this paper, we propose a bidirectional algorithm for sequence labeling to capture the influence of both the left-to-right and the right-to-left directions. We combine the optimization of two unidirectional models from opposite directions via the dual decomposition method to jointly label the input sequence. Experiments on three sequence labeling tasks (Chinese word segmentation, English POS tagging and text chunking) show that our approach can improve the accuracy of sequence labeling tasks when the two unidirectional models individually make highly different predictions.

1 Introduction

Many natural language processing tasks, e.g., POS tagging, text chunking and Chinese word segmentation, can be formulated as a sequence labeling problem. In these tasks, each token in a sequence is assigned a label, and the label assignment of a given token is influenced by the label assignments of the previous tokens. Most sequence labeling models are unidirectional where the inference procedure is performed in one direction only (left to right, or right to left, but not both). As a result, only the influence of one direction is explicitly considered. For many sequence labeling tasks, however, both the left and right contexts can be useful and should be taken into account. For example, consider the POS tagging procedure for the sentence "Would service be voluntary or compulsory?". The word "service" can either be labeled as a verb or a noun. In a left-to-right model, the POS tag "MD" of the previous word "Would" strongly indicates that "service" should be tagged as verb. However, this is the incorrect answer in the case. In a right-to-left model, the POS tag "VB" of the following word "be" indicates "service" should be a noun, which is the correct answer. This means that a model that accounts for the influence of both the left and right contexts is better.

In recent years, a number of bidirectional sequence labeling models were proposed to exploit the influence of both directions. Liu and Zong (2003) and Shen et al. (2003) improved the tagging accuracy by pairwise combining or voting between the left-to-right and right-to-left taggers. Toutanova et al. (2003) proposed a POS tagging model

based on bidirectional dependency networks that make the right context available for a left-to-right model. Tsuruoka and Tsujii (2005) considered all possible decompositions of bidirectional contexts, and chose one that has the highest probability among different taggers. Shen et al. (2007) extended Tsuruoka and Tsujii (2005) and integrated the inference order selection and classifier training into a single learning framework.

In this paper, we propose a novel approach for bidirectional sequence labeling. We combine the optimization of two unidirectional models from opposite directions to predict agreed labels through the dual decomposition method. We estimated our approach on three sequence labeling tasks for two languages: Chinese word segmentation, English POS tagging and text chunking. Experimental results show that our approach is effective when the two unidirectional models individually make highly different predictions.

2 Unidirectional Approach

Let us denote the input sequence of tokens as $x = x_1 x_2 \dots x_n$, and the label sequence for *x* as $y = y_1 y_2 \dots y_n$, where y_i (belonging to a label set $\boldsymbol{Y}$) is the label for the token x_i. For example, in part-of-speech tagging, the input sequence would be the word tokens in a sentence and the output would be POS tags for the word tokens.

The task of sequence labeling is to find the best label sequence $\hat{y}$ for an input sequence *x*:

$$\hat{y} = \underset{y \in \mathcal{Y}}{argmax}\, p(y|x) \quad (1)$$

Usually, the global probability $p(y|x)$ can be decomposed into products of a sequence of local predictions. For example, in the left-to-right model, the probability is decomposed into:

$$p(y|x) = \prod_{i=1}^{n} p(y_i|x, y_1 \dots y_{i-1}) \quad (2)$$

where $p(y_i|x, y_1 \dots y_{i-1})$ is the prediction probability of assigning y_i for x_i. Here, we model the prediction probability with the Maximum Entropy (ME) model:

$$p(y_i|x, y_1 \dots y_{i-1}) = \frac{exp(w \cdot \phi(x, y_1 \dots y_{i-1}, y_i))}{\sum_{y_i'} exp(w \cdot \phi(x, y_1 \dots y_{i-1}, y_i'))} \quad (3)$$

where $\phi(x, y_1 \dots y_{i-1}, y_i)$ is a feature vector, and w is the weight vector for those features. When given a training set of labeled sequences, we can estimate the model parameter *w* using the usual way for ME models, i.e., Generalized Iterative Scaling (GIS) or gradient descent methods.

The probability of the current label prediction in E.q (3) is conditioned on label predictions for previous tokens. If we make a first-order Markov assumption, the Viterbi algorithm would be an efficient decoding method. However, Jiang et al., (2008) showed that non-local features are much helpful for POS tagging. Therefore, we design a unidirectional decoding algorithm that uses more than one prediction before the current position.

Algorithm 1 shows the decoding algorithm, which is based on the beam search algorithm. We use two max-heaps to hold the partial label sequences, where *preHeap* maintains a list of *N* best partial candidates ending at position *i-1* and *curHeap* maintains a list of *N* best partial candidates ending at position *i*. The algorithm initializes the

preHeap with an empty sequence (line 1). It then traverses the input sequence from left to right, and assigns a label to each token (line 2 to line 13). When processing the *i*-th token x_i, the algorithm extracts the top partial candidate *item* from *preHeap* (line 6), and tries to extend *item* with each label in the label set $\boldsymbol{Y}$. If a label y_i is compatible with *item* (line 9), we build a new partial candidate *item'* by combining y_i with *item* (line 11), calculate the probability of *item'* using E.q. (2) (line 10) and add it to *curHeap* (line 12). When all the input tokens are processed, the best partial candidate in *curHeap* is returned as the final result (line 14).

Algorithm 1 Unidirectional Decoding Algorithm

Input: sequence $x = x_1 \dots x_n$, beam size *N*
Output: label sequence $y = y_1 y_2 \dots y_n$
1: *preHeap* ←New-Item(*null*)
2: **for** *i* ← *1…n* **do**
3: *curHeap* ← ∅
4: *k* ← *0*
5: **while** | *preHeap* | > 0 and *k* < *N* **do**
6: *item* ← Pop-Max(*preHeap*)
7: *k* ← *k* + *1*
8: **for** y_i in $\boldsymbol{Y}$ **do**
9: **if** IsCompatible (*item*, y_i, x_i) **then**
10: *prob* ← Eval(*i*, *x*, *item*, y_i)
11: *item'* ← New-Item(*item*, y_i,*prob*)
12: Push(*curHeap*, *item'*)
13: *preHeap* ← *curHeap*
14: **return** Pop-Max(*curHeap*)

Although the model and the decoding algorithm are designed for the left-to-right direction, they can be trivially adapted to the right-to-left direction. To train a right-to-left model, we just reverse all the label sequences in the training set before training. For decoding, we reverse the input sequence first, then decode the reversed sequence with the right-to-left model and reverse the label sequence back.

3 Bidirectional Decoding

In this section, we describe how to improve sequence labeling by jointly optimizing the two unidirectional models. We train a left-to-right model and a right-to-left model and then jointly label an input sequence with the two models.

For purposes of clarity, we define some notations first. The label sequence from the left-to-right model is denoted as $l = l_1 l_2 \dots l_n$, and the output from the right-to-left model is denoted as $r = r_1 r_2 \dots r_n$. For $l = l_1 l_2 \dots l_n$, we define $l(i,t) = 1$ if l_i is assigned with a label $t \in Y$, otherwise $l(i,t) = 0$. Similarly, for $r = r_1 r_2 \dots r_n$, we define $r(i,t) = 1$ if r_i is assigned with a label $t \in Y$, otherwise $r(i,t) = 0$. Therefore, l and r are equal, only if $l(i,t) = r(i,t)$ for all $i \in [1,n]$ and $t \in Y$, otherwise they are unequal.

We expect the two unidirectional models to predict equal results and formulate it as a constraint optimization problem:

$$(\hat{l}, \hat{r}) = \underset{l,r}{\text{argmax}}\ f_1(l) + f_2(r)$$

Such that for all $i \in [1, n]$ and $t \in Y$: $l(i,t) = r(i,t)$

where $f_1(l) = log\ p(l|x) = \sum_{i=1}^{n} \log p(l_i|x, l_1 \dots l_{i-1})$ is a score estimated from the left-to-right model, and $f_2(r) = log\ p(r|x)$ is a score estimated from the right-to-left model.

The dual decomposition (a special case of Lagrangian relaxation) method introduced in Rush et al. (2010) is suitable for this problem. Following their method, we solve the primal constraint optimization problem by optimizing the *dual* problem. First, we introduce a vector of Lagrange multiplier $\mu(i,t)$ for each equality constraint: $l(i,t) = r(i,t)$. Then, the Lagrangian is formulated as:

$$L(l, r, \mu) = f_1(l) + f_2(r) + \sum_{i,t} \mu(i,t)(l(i,t) - r(i,t))$$

By grouping the terms that depend on l and r, we rewrite the Lagrangian as

$$L(l, r, \mu) = \left(f_1(l) + \sum_{i,t} \mu(i,t)s(i,t)\right) + \left(f_2(r) - \sum_{i,t} \mu(i,t)t(i,t)\right)$$

Then, the *dual objective* is

$$\begin{aligned} L(\mu) &= \max_{l,r} L(l, r, \mu) \\ &= \max_{l}\left(f_1(l) + \sum_{i,t} \mu(i,t)l(i,t)\right) \\ &\quad + \max_{r}\left(f_2(r) - \sum_{i,t} \mu(i,t)r(i,t)\right) \end{aligned}$$

The dual problem is to find the $\min_{\mu} L(\mu)$.

We use the subgradient method (Boyd et al., 2003) to minimize the dual. Following Rush et al. (2010), we define the subgradient of $L(\mu)$ as:

$$\gamma(i,t) = l(i,t) - r(i,t) \text{ for all } (i,t).$$

Then, adjust $\mu(i,t)$ as follows:

$$\mu'(i,t) = \mu(i,t) - \delta(l(i,t) - r(i,t))$$

where δ>0 is a step size.

Algorithm 2 Bidirectional Decoding Algorithm

1: Set $\mu^{(0)}(i,t)$=0, for all $i \in [1,n]$ and $t \in \boldsymbol{Y}$
2: **for** k = 1 **to** K **do**
3: $\quad \hat{l}^{(k)} \leftarrow \underset{l}{\text{argmax}}\left(f_1(l) + \sum_{i,t} \mu^{(k-1)}(i,t)l(i,t)\right)$
4: $\quad \hat{r}^{(k)} \leftarrow \underset{r}{\text{argmax}}\left(f_2(r) - \sum_{i,t} \mu^{(k-1)}(i,t)r(i,t)\right)$
5: $\quad$ **if** $l^{(k)}(i,t) = r^{(k)}(i,t)$ for all (i,t) **then**
6: $\quad\quad$ **return** $(\hat{l}^{(k)}, \hat{r}^{(k)})$
7: $\quad$ **else**
8: $\quad\quad \mu^{(k)}(i,t) = \mu^{(k-1)}(i,t) - \delta\left(l^{(k)}(i,t) - r^{(k)}(i,t)\right)$

Algorithm 2 presents the subgradient method to solve the dual problem. The algorithm initializes the Lagrange multiplier values with 0 (line 1) and then iterates many times. At each iteration, the algorithm finds the best $\hat{l}^{(k)}$ and $\hat{r}^{(k)}$ through the left-to-

right model (line 3) and the right-to-left model (line 4) individually. If $\hat{l}^{(k)}$ and $\hat{r}^{(k)}$ are equal (line 5), then the algorithm returns the solution (line 6). Otherwise, the algorithm adjusts the Lagrange multiplier values based on the differences between $\hat{l}^{(k)}$ and $\hat{r}^{(k)}$ (line 8). A crucial point is that the argmax problems in line 3 and line 4 can be solved efficiently using the original unidirectional decoding algorithms, because the Lagrange multiplier can be regarded as adjustments for the prediction score $\log p(y_i|x, y_1 \ldots y_{i-1})$ of each token. According to the strong duality theorem (Korte and Vygen, 2008), the dual solution is the label sequence we want to get.

4 Experiment

To evaluate the effectiveness of our method, we conducted experiments on three sequence labeling tasks: Chinese word segmentation, English POS tagging and text chunking.

4.1 Tasks and Data Sets

The task of Chinese word segmentation is segmenting a sequence of Chinese characters into words. The character-based model (Xue, 2003) treats segmentation as a sequence labeling task, where each Chinese character is labeled with a tag. We used the tag set used in Wang et al. (2011). We split the Chinese Treebank Version 5.0 (CTB5) with the standard data split: 1-270, 400-1151 as the training set, 301-325 as the development set and 271-300 as the test set.

We split the Penn Wall Street Journal Treebank (WSJ) with the standard data split for POS tagging: sections 0-18 as the training set, sections 19-21 as the development set and sections 22-24 as the test set.

The task of text chunking is to find non-recursive phrases in a sentence. We treat it as a tagging task by converting chunks into tags on tokens. We choose the IOB scheme: each token gets the label B-X if it is the first token in chunk X, the label I-X if it is not the first token in chunk X, or the label O if it is outside of any chunks. We used the data set from the CoNLL-2000 shared task.

The feature templates for each task are adopted from previous work. For Chinese word segmentation, we use the feature templates provided in Wang et al. (2011). For POS tagging and chunking, we used the feature templates provided in Tsuruoka and Tsujii (2005), excluding those temples containing future predictions.

4.2 Results

We built three systems for each task. The “left-to-right” system and the “right-to-left” system were two unidirectional systems, which trained models and decoded sequences from opposite directions. The “bidirectional” system used these two unidirectional models jointly to decode sequences with Algorithm 2. We trained models for three tasks with the Maximum Entropy model implemented in the OpenNLP toolkit.

We tuned parameters on the development set and finally set the beam size (in Algorithm 1) to N=20, the maximum iteration to K=30 and the step size to δ=0.5 (in Algorithm 2). The experimental results on the test set are presented in Table 1 and they show that the accuracy of the POS tagging task and the F1 score of the chunking task were improved when using the bidirectional decoding algorithm. However, the Chinese word segmentation task showed no improvement.

Fig. 1 illustrates how the bidirectional de-coding algorithm leads to improvement over unidirectional models when assigning POS tags to the sentence "Would service be voluntary or compulsory?". In the left-to-right model, the word token "service" is labeled with an erroneous tag "VB", because the preceding word "Would" is a modal verb that is often followed by a verb. In the right-to-left model, "service" is correctly labeled, because the following word "be" is a verb that is often preceded by nouns. However, the right-to-left model assigns the wrong tag "NN" to the word "compulsory", presumably because it is the first token in the sequence and "NN" is a more likely tag for the first token. The left-to-right model, on the other hand, assigns the correct label "JJ". The bidirectional algorithm combines the strengths of both models and assigns the correct tags to all words.

		F1(%)
Chinese Word Segmentation	left-to-right	97.67
	right-to-left	97.55
	bidirectional	97.65
		Accuracy(%)
POS Tagging	left-to-right	96.83
	right-to-left	96.84
	bidirectional	**97.15**
		F1(%)
Chunking	left-to-right	93.42
	right-to-left	93.37
	bidirectional	**93.61**

Table 1. Experimental results on the test set.

	Would	service	be	voluntary	or	compulsory	?
Gold	MD	NN	VB	JJ	CC	JJ	.
Left-to-right:	MD	VB	VB	JJ	CC	JJ	.
Right-to-left:	MD	NN	VB	JJ	CC	NN	.
Bidirectional:	MD	NN	VB	JJ	CC	JJ	.

Fig. 1. A POS tagging example, where the wrong tags are highlighted with red color.

4.3 Discussion

To understand the scenarios where the bidirectional decoding algorithm is effective, we analyzed the three tasks in detail. Table 2 presents the total number of tokens in the test set and the number of tokens to which the left-to-right and right-to-left models assigned different labels. We found the number of tokens receiving different labels was low for

the Chinese word segmentation task, but high for the English POS tagging and chunking tasks. Combined with the results in Table 1, we can conclude that our algorithm is effective when the two unidirectional models make very different predictions. When the two unidirectional models make the same predictions, even if the predictions are wrong, the bidirectional algorithm can do nothing to correct them.

We also estimated the convergence of the bidirectional decoding algorithm by counting the number of iterations when the two unidirectional models make different predictions. Fig. 2 shows the percentage of sequences where exact solutions are returned versus the number of iterations. We find our algorithm produces exact solutions to over 80% of the sequences within 10 iterations.

	Total Tokens	Inconsistent Tokens
Word Seg.	13,738	48
POS Tagging	129,654	2,384
Chunking	47,377	980

Table 2. Differences between the left-to-right and the right-to-left results.

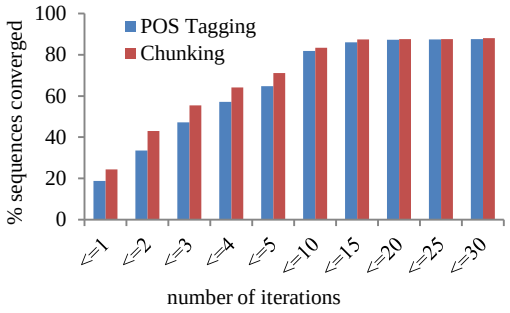

Fig. 2. Convergence of the bidirectional decoding algorithm.

5 Conclusion

In this paper, we proposed a bidirectional decoding algorithm for sequence labeling tasks. We use two unidirectional models of opposite directions to jointly label the input sequences via the dual decomposition algorithm. Experiments on three sequence labeling tasks show that our approach improves the performance on sequence labeling tasks when the two unidirectional models makes very different predictions.

Acknowledgments

The research work has been funded by the Hi-Tech Research and Development Program ("863" Program) of China under Grant No. 2011AA01A207, 2012AA011101,

and 2012AA011102. This work is also supported in part by the DAPRA via contract HR0011-11-C-0145 entitled "Linguistic Resources for Multilingual Processing".

References

S. Boyd, L. Xiao and A. Mutapcic. 2003. Subgradient methods. Lecture notes of EE392o, Stanford University.

D. Liu and C. Zong. 2003. Utterance Segmentation Using Combined Approach Based on Bi-directional N-gram and Maximum Entropy. In ACL-2003 Workshop: The Second SIGHAN Workshop on Chinese Language Processing.

W. Jiang, H. Mi and Q. Liu. 2008. Word lattice reranking for Chinese word segmentation and part-of-speech tagging. In Coling 2008.

B. Korte and J. Vygen. 2008. Combinatorial optimization: theory and algorithms: Springer.

A. M. Rush, D. Sontag, M. Collins and T. Jaakkola. 2010. On dual decomposi-tion and linear programming relaxations for natural language processing. In EMNLP2010.

L. Shen and A. K. Joshi. 2003. A SNoW based Supertagger with Application to NP Chunking. In ACL 2003.

L. Shen, G.Satta and A. K. Joshi. 2007. Guided Learning for Bidirectional Sequence Classification. In ACL2007.

K. Toutanova, D. Klein, C. Manning and Y. Singer. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In NAACL 2003.

Y. Tsuruoka and J. Tsujii. 2005. Bidirectional inference with the easiest-first strategy for tagging sequence data. In EMNLP2005.

Y. Wang, J. Kazama, Y. Tsuruoka, W. Chen, Y. Zhang and K. Torisawa. 2011. Improving Chinese word segmentation and POS tagging with semi-supervised methods using large auto-analyzed data. In IJCNLP2011.

N. Xue. 2003. Chinese word segmentation as character tagging. Computational Linguistics and Chinese Language Processing, 8 (1). pages 29-48.