Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16)

Exploring Segment Representations for Neural Segmentation Models

Yijia Liu, Wanxiang Che *, Jiang Guo, Bing Qin, Ting Liu
Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology, China
{yjliu,car,jguo,qinb,tliu}@ir.hit.edu.cn

Abstract

Many natural language processing (NLP) tasks can be generalized into segmentation problem. In this paper, we combine semi-CRF with neural network to solve NLP segmentation tasks. Our model represents a segment both by composing the input units and embedding the entire segment. We thoroughly study different composition functions and different segment embeddings. We conduct extensive experiments on two typical segmentation tasks: named entity recognition (NER) and Chinese word segmentation (CWS). Experimental results show that our neural semi-CRF model benefits from representing the entire segment and achieves the state-of-the-art performance on CWS benchmark dataset and competitive results on the CoNLL03 dataset.

1 Introduction

Given an input sequence, *segmentation* is the problem of identifying and assigning tags to its subsequences. Many natural language processing (NLP) tasks can be cast into the segmentation problem, like named entity recognition [Okanohara *et al.*, 2006], opinion extraction [Yang and Cardie, 2012], and Chinese word segmentation [Andrew, 2006]. Properly representing *segment* is critical for good segmentation performance. Widely used sequence labeling models like conditional random fields [Lafferty *et al.*, 2001] represent the contextual information of the segment boundary as a proxy to entire segment and achieve segmentation by labeling input units (e.g. words or characters) with boundary tags. Compared with sequence labeling model, models that directly represent segment are attractive because they are not bounded by local tag dependencies and can effectively adopt segment-level information. Semi-Markov CRF (or semi-CRF) [Sarawagi and Cohen, 2004] is one of the models that directly represent the entire segment. In semi-CRF, the conditional probability of a semi-Markov chain on the input sequence is explicitly modeled, whose each state corresponds to a subsequence of input units, which makes semi-CRF a natural choice for segmentation problem.

However, to achieve good segmentation performance, conventional semi-CRF models require carefully hand-crafted features to represent the segment. Recent years witness a trend of applying neural network models to NLP tasks. The key strengths of neural approaches in NLP are their ability for modeling the compositionality of language and learning distributed representation from large-scale unlabeled data. Representing a segment with neural network is appealing in semi-CRF because various neural network structures [Hochreiter and Schmidhuber, 1997] have been proposed to compose sequential inputs of a segment and the well-studied word embedding methods [Mikolov *et al.*, 2013] make it possible to learn entire segment representation from unlabeled data.

In this paper, we combine neural network with semi-CRF and make a thorough study on the problem of representing a segment in neural semi-CRF. Kong *et al.* [2015] proposed a segmental recurrent neural network (SRNN) which represents a segment by composing input units with RNN. We study alternative network structures besides the SRNN. We also study segment-level representation using *segment embedding* which encodes the entire segment explicitly. We conduct extensive experiments on two typical NLP segmentation tasks: named entity recognition (NER) and Chinese word segmentation (CWS). Experimental results show that our concatenation alternative achieves comparable performance with the original SRNN but runs 1.7 times faster and our neural semi-CRF greatly benefits from the segment embeddings. In the NER experiments, our neural semi-CRF model with segment embeddings achieves an improvement of 0.7 F-score over the baseline and the result is competitive with state-of-the-art systems. In the CWS experiments, our model achieves more than 2.0 F-score improvements on average. On the PKU and MSR datasets, state-of-the-art F-scores of 95.67% and 97.58% are achieved respectively. We release our code at https://github.com/ExpResults/segrep-for-nn-semicrf.

2 Problem Definition

Figure 1 shows examples of named entity recognition and Chinese word segmentation. For the input word sequence in the NER example, its segments (*"Michael Jordan":PER, "is":NONE, "a":NONE, "professor":NONE, "at":NONE, "Berkeley":ORG*) reveal that "Michaels Jordan" is a person name and "Berkeley" is an organization. In the CWS example, the subsequences ("浦东/Pudong", "开发/development",

*Email corresponding.

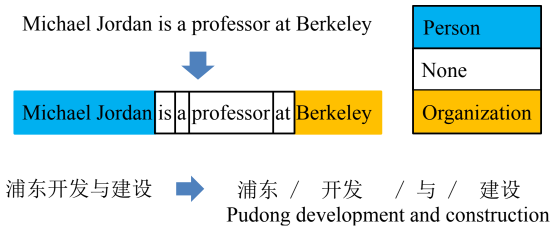


Figure 1: Examples for named entity recognition (above) and Chinese word segmentation (below).

"与/and", "建设/construction") of the input character sequence are recognized as words. Both NER and CWS take an input sequence and partition it into disjoint subsequences.

Formally, for an input sequence $\mathbf{x} = (x_1, .., x_{|\mathbf{x}|})$ of length $|\mathbf{x}|$, let $x_{a:b}$ denote its subsequence $(x_a, ..., x_b)$. A *segment* of $\mathbf{x}$ is defined as (u, v, y) which means the subsequence $x_{u:v}$ is associated with label y. A *segmentation* of $\mathbf{x}$ is a *segment* sequence $\mathbf{s} = (s_1, .., s_p)$, where $s_j = (u_j, v_j, y_j)$ and $u_{j+1} = v_j + 1$. Given an input sequence $\mathbf{x}$, the *segmentation* problem can be defined as the problem of finding $\mathbf{x}$'s most probable *segment* sequence $\mathbf{s}$.

3 Neural Semi-Markov CRF

Semi-Markov CRF (or semi-CRF, Figure 2a) [Sarawagi and Cohen, 2004] models the conditional probability of $\mathbf{s}$ on $\mathbf{x}$ as

$$p(\mathbf{s}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\{W \cdot G(\mathbf{x}, \mathbf{s})\}$$

where $G(\mathbf{x}, \mathbf{s})$ is the feature function, W is the weight vector and $Z(\mathbf{x}) = \sum_{\mathbf{s}' \in \mathbf{S}} \exp\{W \cdot G(\mathbf{x}, \mathbf{s}')\}$ is the normalize factor of all possible *segmentations* $\mathbf{S}$ over $\mathbf{x}$.

By restricting the scope of feature function within a segment and ignoring label transition between segments (0-order semi-CRF), $G(\mathbf{x}, \mathbf{s})$ can be decomposed as $G(\mathbf{x}, \mathbf{s}) = \sum_{j=1}^{p} g(\mathbf{x}, s_j)$ where $g(\mathbf{x}, s_j)$ maps segment s_j into its representation. Such decomposition allows using efficient dynamic programming algorithm for inference. To find the best segmentation in semi-CRF, let α_j denote the best segmentation ends with j^{th} input and α_j is recursively calculated as

$$\alpha_j = \max_{l=1..L,y} \Psi(j - l, j, y) + \alpha_{j-l-1}$$

where L is the maximum length manually defined and $\Psi(j - l, j, y)$ is the transition weight for $s = (j - l, j, y)$ in which $\Psi(j - l, j, y) = W \cdot g(\mathbf{x}, s)$.

Previous semi-CRF works [Sarawagi and Cohen, 2004; Okanohara *et al.*, 2006; Andrew, 2006; Yang and Cardie, 2012] parameterize $g(\mathbf{x}, s)$ as a sparse vector, each dimension of which represents the value of corresponding feature function. Generally, these feature functions fall into two types: 1) the *CRF style features* which represent input unit-level information such as "the specific words at location i" 2) the *semi-CRF style features* which represent segment-level information such as "the length of the segment".

Kong *et al.* [2015] proposed the segmental recurrent neural network model (SRNN, see Figure 2b) which combines the semi-CRF and the neural network model. In SRNN, $g(\mathbf{x}, s)$ is parameterized as a bidirectional LSTM (bi-LSTM). For a segment $s_j = (u_j, v_j, y_j)$, each input unit x in subsequence $(x_{u_j}, .., x_{v_j})$ is encoded as *embedding* and fed into the bi-LSTM. The rectified linear combination of the final hidden layers from bi-LSTM is used as $g(\mathbf{x}, s)$. Kong *et al.* [2015] pioneers in representing a segment in neural semi-CRF. Bi-LSTM can be regarded as "neuralized" *CRF style features* which model the input unit-level compositionality. However, in the SRNN work, only the bi-LSTM was employed without considering other input unit-level composition functions. What is more, the *semi-CRF styled* segment-level information as an important representation was not studied. In the following sections, we first study alternative input unit-level composition functions (3.1). Then, we study the problem of representing a segment at segment-level (3.2).

3.1 Alternative Seg-Rep. via Input Composition

Segmental CNN

Besides recurrent neural network (RNN) and its variants, another widely used neural network architecture for composing and representing variable-length input is the convolutional neural network (CNN) [Collobert *et al.*, 2011]. In CNN, one or more filter functions are employed to convert a fix-width segment in sequence into one vector. With filter function "sliding" over the input sequence, contextual information is encoded. Finally, a pooling function is used to merge the vectors into one. In this paper, we use a filter function of width 2 and max-pooling function to compose input units of a segment. Following SRNN, we name our CNN segment representation as SCNN (see Figure 2c).

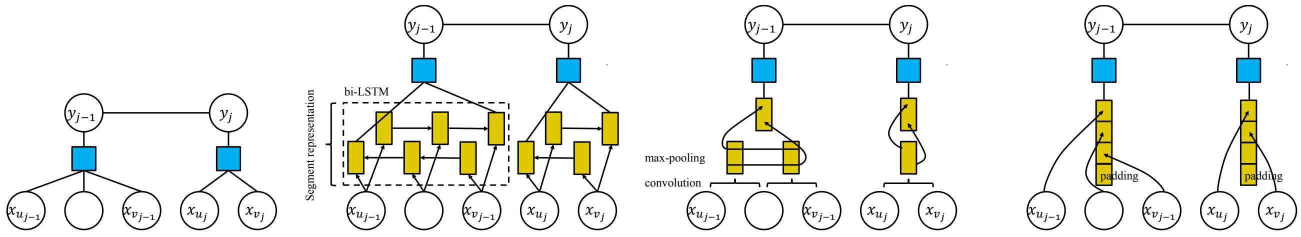
However, one problem of using CNN to compose input units into segment representation lies in the fact that the max-pooling function is insensitive to input position. Two different segments sharing the same vocabulary can be treated without difference. In a CWS example, "球拍卖" (racket for sell) and "拍卖球" (ball audition) will be encoded into the same vector in SCNN if the vector of "拍卖" that produced by filter function is always preserved by max-pooling.

Segmental Concatenation

Concatenation is also widely used in neural network models to represent fixed-length input. Although not designed to handle variable-length input, we see that in the inference of semi-CRF, a maximum length L is adopted, which make it possible to use padding technique to transform the variable-length representation problem into fixed-length of L. Meanwhile, concatenation preserves the positions of inputs because they are directly mapped into the certain positions in the resulting vector. In this paper, we study an alternative concatenation function to compose input units into segment representation, namely the SCONCATE model (see Figure 2d). Compared with SRNN, SCONCATE requires less computation when representing one segment, thus can speed up the inference.

3.2 Seg-Rep. via Segment Embeddings

For segmentation problems, a segment is generally considered more informative and less ambiguous than an individual input. Incorporating segment-level features usually lead



(a) semi-CRF (b) SRNN (c) SCNN (d) SCONCATE

Figure 2: An illustration for the semi-CRF, SRNN, SCNN and SCONCATE. In these figures, circles represent the inputs, blue rectangles represent *factors* in graphic model and yellow rectangles represent generic nodes in the neural network model.

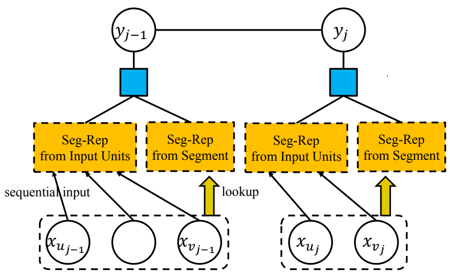


Figure 3: Our neural semi-CRF model with segment representation from input composition and segment embeddings.

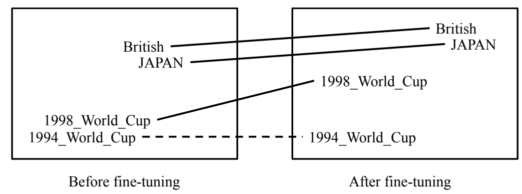


Figure 4: An example for fine-tuning decreases the generalization power of pre-trained segment embedding. “1994_World_Cup” does not occur in the training data and its similarity with “1998_World_Cup” is broken because “1998_World_Cup” is tuned.

performance improvement in previous semi-CRF work. Segment representations in Section 3.1 only model the composition of input units. It can be expected that the segment embedding which encodes an entire subsequence as a vector can be an effective way for representing a segment.

In this paper, we treat the segment embedding as a lookup-based representation, which retrieves the embedding table with the surface string of entire segment. With the entire segment properly embed, it is straightforward to combine the segment embedding with the composed vector from the input so that multi-level information of a segment is used in our model (see Figure 3). However, how to obtain such embeddings is not a trivial problem.

A natural solution for obtaining the segment embeddings can be collecting all the “correct” segments from training data into a lexicon and learning their embeddings as model parameters. However, the in-lexicon segment is a strong clue for a subsequence being a correct segment, which makes our model vulnerable to overfitting. Unsupervised pre-training has been proved an effective technique for improving the robustness of neural network model [Erhan *et al.*, 2010]. To mitigate the overfitting problem, we initialize our segment embeddings with the pre-trained one.

Word embedding gains a lot of research interest in recent years [Mikolov *et al.*, 2013] and is mainly carried on English texts which are naturally segmented. Different from the word embedding works, our segment embedding requires large-scale segmented data, which cannot be directly obtained. Following Wang *et al.* [2011] which utilize automatically segmented data to enhance their model, we obtain the auto-segmented data with our neural semi-CRF baselines (SRNN, SCNN, and SCONCATE) and use the auto-segmented data to learn our segment embeddings.

Another line of research shows that machine learning algorithms can be boosted by ensembling *heterogeneous* models. Our neural semi-CRF model can take knowledge from heterogeneous models by using the segment embeddings learned on the data segmented by the heterogeneous models. In this paper, we also obtain the auto-segmented data from a conventional CRF model which utilizes hand-crafted sparse features. Once obtaining the auto-segmented data, we learn the segment embeddings in the same with word embeddings.

A problem that arises is the fine-tuning of segment embeddings. Fine-tuning can learn a task-specific segment embeddings for the segments that occur in the training data, but it breaks their relations with the un-tuned out-of-vocabulary segments. Figure 4 illustrates this problem. Since OOV segments can affect the testing performance, we also try learning our model without fine-tuning the segment embeddings.

3.3 Model details

In this section, we describe the detailed architecture for our neural semi-CRF model.

Input Unit Representation

Following Kong *et al.* [2015], we use a bi-LSTM to represent the input sequence. To obtain the input unit representation, we use the technique in Dyer *et al.* [2015] and separately use two parts of input unit embeddings: the pre-trained embeddings E^p without fine-tuning and fine-tuned embeddings E^t. For the ith input, E^p_i and E^t_i are merged together through linear combination and form the input unit representation

$$I_i = \text{ReLU}(W^{\mathcal{I}}[E^p_i; E^t_i] + b^{\mathcal{I}})$$

fixed input unit embedding E_i^p size	100
fine tuned input unit embedding E_i^t size	32
input unit representation I_i size	100
LSTM hidden layer H_i size	100
seg-rep via input composition SCOMP	64
seg-rep via segment embedding SEMB	50
label embedding $E_{y_i}^Y$ size	20
final segment representation S_i size	100

Table 1: Hyper-parameter settings

where the notation of $W[X_1; ..; X_n]$ equals to $X_1, .., X_n$'s linear combination $W_1X_1 + .. + W_nX_n$ and b^I is the bias. After obtaining the representation for each input unit, a sequence $(I_1, ..., I_{|\mathbf{x}|})$ is fed to a bi-LSTM. The hidden layer of forward LSTM $\overrightarrow{H_i}$ and backward LSTM $\overleftarrow{H_i}$ are combined as

$$H_i = \text{ReLU}(W^{\mathcal{H}}[\overrightarrow{H_i}; \overleftarrow{H_i}] + b^{\mathcal{H}})$$

and used as the i^{th} input unit's final representation.

Segment Representation

Given a segment $s_j = (u_j, v_j, y_j)$, a generic function $\text{SComp}(H_{u_j}, ..., H_{v_j})$ stands for the segment representation that composes the input unit representations $(H_{u_j}, .., H_{v_j})$. In this work, SCOMP is instantiated with three different functions: SRNN, SCNN and SCONCATE. Besides composing input units, we also employ the segment embeddings as segment-level representation. Embedding of the segment $s_j = (u_j, v_j, y_j)$ is denoted as a generic function $\text{SEmb}(x_{u_j} ... x_{v_j})$ which converts the subsequence $(x_{u_j}, ..., x_{v_j})$ into its embedding through a lookup table. At last, the representation of segment s_j is calculated as

$$S_j = \text{ReLU}(W^{\mathcal{S}}[\text{SComp}_j; \text{SEmb}_j; E_{y_j}^Y] + b^{\mathcal{S}})$$

where E^Y is the embedding for the label of a segment.

Throughout this paper, we use the same hyper-parameters for different experiments as listed in Table 1.

Training Procedure

In this paper, negative log-likelihood is used as learning objective. We follow Dyer *et al.* [2015] and use stochastic gradient descent to optimize model parameters. Initial learning rate is set as $\eta_0 = 0.1$ and updated as $\eta_t = \eta_0/(1 + 0.1t)$ on each epoch t. Best training iteration is determined by the evaluation score on development data.

4 Experiment

We conduct our experiments on two NLP segmentation tasks: named entity recognition and Chinese word segmentation.

4.1 Dataset and Word Embedding

For NER, we use the CoNLL03 dataset which is widely adopted for evaluating NER models' performance. F-score is used as evaluation metric.[1]

For CWS, we follow previous study and use three Simplified Chinese datasets: PKU and MSR from 2nd SIGHAN bakeoff and Chinese Treebank 6.0 (CTB6). For the PKU and MSR datasets, last 10% of the training data are used as development data as [Pei *et al.*, 2014] does. For CTB6 data, recommended data split is used. We convert all the double byte digits and letters in the PKU data into single byte. Like NER, CWS performance is evaluated by F-score.[2]

Unlabeled data are used to learn both the input unit embeddings (word embedding for NER, character embedding for CWS) and segment embeddings. For NER, we use RCV1 data as our unlabeled English data. For CWS, Chinese gigawords is used as unlabeled Chinese data. Throughout this paper, we use the word embedding toolkit released by Ling *et al.* [2015] to obtain both the input unit embeddings and segment embeddings.[3]

4.2 Baseline

We compare our models with three baselines:

1. SPARSE-CRF: The CRF model using sparse handcrafted features.
2. NN-LABELER: The neural network sequence labeling model making classification on each input unit.
3. NN-CRF: The neural network CRF which models the conditional probability of a label sequence over the input sequence.

BIESO-tag schema is used in all the CRF and sequence labeling models.[4] For SPARSE-CRF, we use the baseline feature templates in Guo *et al.* [2014] for NER and Jiang *et al.* [2013]'s feature templates for CWS. Both NN-LABELER and NN-CRF take the same input unit representation as our neural semi-CRF models but vary on the output structure and do not explicitly model segment-level information.

4.3 Comparing Different Input Composition Functions

We first consider the problem of representing segments via composing input units and compare different input composition functions. Results on NER and CWS data are shown in Table 2. From this table, the SRNN and SCONCATE achieve comparable results and perform better than the SCNN. Although CNN can model input sequence at any length, its invariance to the exact position can be a flaw in representing segments. The experimental results confirm that and show the importance of properly handling the input position. Considering SCNN's relatively poor performance, we only study SRNN and SCONCATE in the following experiments.

Comparing with NN-LABELER, structure prediction models (NN-CRF and neural semi-CRF) generally achieve better performance. The best structure prediction model outperforms NN-LABELER by 0.4% on NER and 1.11% averagely on CWS according to Table 2. But the difference between the neural structure prediction models is not significant. NN-CRF performs better than the best neural semi-CRF model

[1] `conlleval` script in CoNLL03 shared task is used.

[2] `score` script in 2nd SIGHAN bakeoff is used.

[3] https://github.com/wlin12/wang2vec

[4] O tag which means OUTSIDE is not adopted in CWS experiments since CWS doesn't involve assigning tags to segments.

		NER		CWS						
		CoNLL03		CTB6		PKU		MSR		
	model	dev	test	dev	test	dev	test	dev	test	spd
baseline	NN-LABELER	93.03	88.62	93.70	93.06	93.57	92.99	93.22	93.79	**3.30**
	NN-CRF	**93.06**	**89.08**	94.33	93.65	94.09	93.28	93.81	94.17	2.72
	SPARSE-CRF	88.87	83.43	**95.68**	**95.08**	**95.85**	**95.06**	**96.09**	**96.54**	
neural semi-CRF	SRNN	92.97	88.63	94.56	94.06	94.86	93.91	94.38	95.21	0.62
	SCONCATE	92.96	89.07	94.34	93.96	94.41	93.57	94.05	94.53	1.08
	SCNN	91.53	87.68	87.82	87.51	79.64	80.75	85.04	85.79	1.46

Table 2: The NER and CWS results of the baseline models and our neural semi-CRF models with different input composition functions. *spd* represents the inference speed and is evaluated by the number of tokens processed per millisecond.

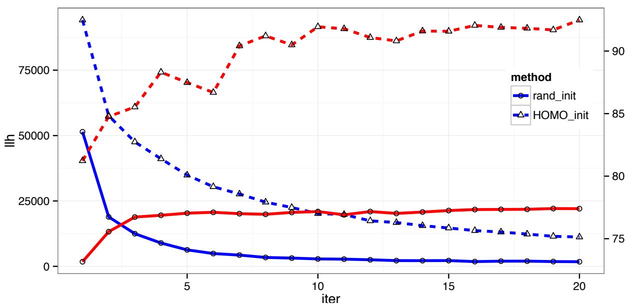


Figure 5: Negative log-likelihood (blue lines) and development F-score (red lines) by iterations. Solid lines show the model with randomly initialized segment embeddings. Dashed lines show that initialized with pre-trained.

(SCONCATE) on NER while the both SRNN and SCONCATE outperform NN-CRF on three CWS datasets. We address this to the fact either the NN-CRF or the neural semi-CRF merely takes input-level information and not sufficiently adopts segment-level information into the models.

A further comparison on inference speed shows that SCONCATE runs 1.7 times faster than SRNN, but slower than the NN-LABELER and NN-CRF, which is resulted from the intrinsic difference in time complexity.

4.4 Comparing Different Segment Embeddings

Next we study the effect of different segment embeddings. Using a segmentation model, we can obtain auto-segmented unlabeled data, then learn the segment embeddings. In this paper, we tried two segmentation models. One is the neural semi-CRF baseline which represents segment by composing input and another one is the CRF model using sparse handcrafted features. For convenience, we use SEMB-HOMO and SEMB-HETERO to note the segment embeddings learned from their auto-segmented data respectively.

Effect of Pre-trained Segment Embeddings

We first incorporate randomly initialized segment embeddings into our model and tune the embeddings along with other parameters. However our preliminary experiments of adding these embeddings into SRNN witness a severe drop of F-score on the CoNLL03 development set (from 92.97% to 77.5%). A further investigation shows that the randomly initialized segment embeddings lead to severe overfitting. Figure 5 shows the learning curve in training the NER model.

model	CoNLL03	CTB6	PKU	MSR
SRNN	92.97	94.56	94.86	94.80
+SEMB-HOMO W/FT	92.97	95.83	**96.70**	**97.32**
+SEMB-HOMO WO/FT	**93.14**	**95.91**	96.64	96.59
SCONCATE	92.96	94.34	94.41	94.05
+SEMB-HOMO W/FT	93.07	95.79	**96.75**	**97.29**
+SEMB-HOMO WO/FT	**93.36**	**95.88**	96.50	96.44
OOV	46.02	5.45	5.80	2.60

Table 3: Effect of fine-tuning (FT) segment embedding on development data. For CoNLL03 data, a named entity is "out-of-vocabulary" when it is not included in the training data as a named entity.

From this figure, the model with randomly initialized segment embeddings converge to the training data at about 5^{th} iteration and the development performance stops increasing at the same time. However, by initializing with SEMB-HOMO, the development set performance increase to 93%, which shows the necessity of pre-trained segment embeddings.

Effect of Fine-tuning Segment Embeddings

We study the effect of fine-tuning the segment embeddings by imposing SEMB-HOMO into our model. Table 3 shows the experimental results on development data. We find that our models benefit from fixing the segment embeddings on CoNLL03. While on MSR, fine-tuning the embeddings helps. Further study on the out-of-vocabulary rate shows that the OOV rate of MSR is very low, thus fine-tuning on segment embeddings help to learn a better task-specified segment representation. However, on CoNLL03 data whose OOV rate is high, fine-tuning the segment embedding harms the generalization power of pre-trained segment embeddings.

Effect of Heterogeneous Segment Embeddings

In previous sections, our experiments are mainly carried on the segment embeddings obtained from homogeneous models. In this section, we use our SPARSE-CRF as the heterogeneous model to obtain SEMB-HETERO. We compare the models with SEMB-HETERO and SEMB-HOMO on the development data in Figure 6. These results show that SEMB-HETERO generally achieve better performance than the SEMB-HOMO. On the CoNLL03 and MSR dataset, the differences are significant. Meanwhile, we see that fine-tuning the segment embedding can narrow the gap between SEMB-HETERO and SEMB-HOMO.

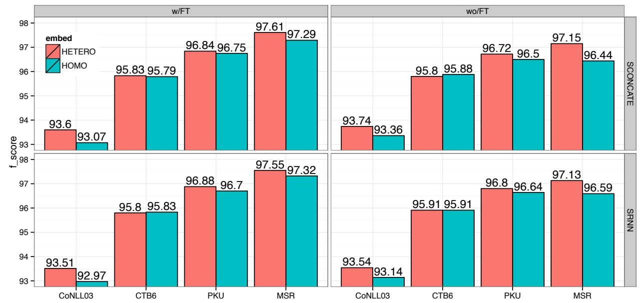


Figure 6: Comparison between models with SEMB-HOMO and SEMB-HETERO on development data. The rows show different baseline neural semi-CRF models and the columns show whether fine-tuning (FT) the segment embeddings.

model	CoNLL03	CTB6	PKU	MSR
NN-LABELER	88.62	93.06	92.99	93.79
NN-CRF	89.08	93.65	93.28	94.17
SPARSE-CRF	83.43	95.08	95.06	96.54
SRNN	88.63	94.06	93.91	95.21
+SEMB-HETERO	89.59	**95.48**	95.60	97.39
	+0.96	*+1.42*	*+1.69*	*+2.18*
SCONCATE	89.07	93.96	93.57	94.53
+SEMB-HETERO	**89.77**	95.42	**95.67**	**97.58**
	+0.70	*+1.43*	*+2.10*	*+3.05*

Table 4: Comparison between baselines and our neural semi-CRF model with segment embeddings.

Final Result

At last, we compare our neural semi-CRF model leveraging additional segment embeddings with those only represent segment by composing input. Table 4 shows the result on the NER and CWS test data. Style of segment embeddings (HOMO or HETERO) and whether fine-tune it is decided by the development data. From this result, we see that segment-level representation greatly boost up model's performance. On NER, an improvement of 0.7% is observed and that improvement on CWS is more than 2.0% on average.

We compare our neural semi-CRF model leveraging multi-levels segment representation with other state-of-the-art NER and CWS systems. Table 5 shows the NER comparison results. The first block shows the results of neural NER models and the second one shows the non-neural models. All these work employed hand-crafted features like capitalization. Collobert *et al.* [2011], Guo *et al.* [2014], and Passos *et al.* [2014] also utilize lexicon as an additional knowledge resource. Without any hand-crafted features, our model can achieve comparable performance with the models utilizing domain-specific features.

Table 6 shows the comparison with the state-of-the-art CWS systems. The first block of Table 6 shows the neural CWS models and second block shows the non-neural models. Our neural semi-CRF model with multi-level segment representation achieves the state-of-the-art performance on PKU and MSR data. On CTB6 data, our model's performance is also close to Wang *et al.* [2011] which uses semi-supervised features extracted auto-segmented unlabeled data. According to Pei *et al.* [2014], significant improvements can be achieved by replacing character embeddings with character-bigram embeddings. However we didn't employ this trick considering the unification of our model.

genre	*model*	CoNLL03
NN	[Collobert *et al.*, 2011]	89.59
	[Huang *et al.*, 2015]	90.10
non-NN	[Ando and Zhang, 2005]	89.31
	[Guo *et al.*, 2014]	88.58
	[Passos *et al.*, 2014]	**90.90**
	our best	89.77

Table 5: Comparison with the state-of-the-art NER systems.

genre	*model*	CTB6	PKU	MSR
NN	[Zheng *et al.*, 2013]	-	92.4	93.3
	[Pei *et al.*, 2014]		94.0	94.9
	[Pei *et al.*, 2014] w/bigram	-	95.2	97.2
	[Kong *et al.*, 2015]		90.6	90.7
non-NN	[Tseng, 2005]	-	95.0	96.4
	[Zhang and Clark, 2007]	-	95.1	97.2
	[Sun *et al.*, 2009]	-	95.2	97.3
	[Wang *et al.*, 2011]	**95.7**	-	-
	our best	95.48	**95.67**	**97.58**

Table 6: Comparison with the state-of-the-art CWS systems.

5 Related Work

Semi-CRF has been successfully used in many NLP tasks like information extraction [Sarawagi and Cohen, 2004], opinion extraction [Yang and Cardie, 2012] and Chinese word segmentation [Andrew, 2006; Sun *et al.*, 2009]. Its combination with neural network is relatively less studied. To the best of our knowledge, our work is the first one that achieves state-of-the-art performance with neural semi-CRF model.

Domain specific knowledge like capitalization has been proved effective in named entity recognition [Ratinov and Roth, 2009]. Segment-level abstraction like whether the segment matches a lexicon entry also leads performance improvement [Collobert *et al.*, 2011]. To keep the simplicity of our model, we didn't employ such features in our NER experiments. But our model can easily take these features and it is hopeful the NER performance can be further improved.

Utilizing auto-segmented data to enhance Chinese word segmentation has been studied in Wang *et al.* [2011]. However, only statistics features counted on the auto-segmented data was introduced to help to determine segment boundary and the entire segment was not considered in their work. Our model explicitly uses the entire segment.

6 Conclusion

In this paper, we systematically study the problem of representing a segment in neural semi-CRF model. We propose a concatenation alternative for representing segment by composing input units which is equally accurate but runs faster than SRNN. We also propose an effective way of incorporating segment embeddings as segment-level representation and it significantly improves the performance. Experiments on named entity recognition and Chinese word segmentation show that the neural semi-CRF benefits from rich segment representation and achieves state-of-the-art performance.

Acknowledgments

This work was supported by the National Key Basic Research Program of China via grant 2014CB340503 and the National Natural Science Foundation of China (NSFC) via grant 61133012 and 61370164.

References

[Ando and Zhang, 2005] Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *J. Mach. Learn. Res.*, 6:1817–1853, December 2005.

[Andrew, 2006] Galen Andrew. A hybrid markov/semi-markov conditional random field for sequence segmentation. In *EMNLP-2006*, pages 465–472, Sydney, Australia, July 2006. ACL.

[Collobert *et al.*, 2011] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, November 2011.

[Dyer *et al.*, 2015] Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. Transition-based dependency parsing with stack long short-term memory. In *ACL-2015*, pages 334–343, Beijing, China, July 2015. ACL.

[Erhan *et al.*, 2010] Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *J. Mach. Learn. Res.*, March 2010.

[Guo *et al.*, 2014] Jiang Guo, Wanxiang Che, Haifeng Wang, and Ting Liu. Revisiting embedding features for simple semi-supervised learning. In *EMNLP-2014*, pages 110–120, Doha, Qatar, October 2014. ACL.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[Huang *et al.*, 2015] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991, 2015.

[Jiang *et al.*, 2013] Wenbin Jiang, Meng Sun, Yajuan Lü, Yating Yang, and Qun Liu. Discriminative learning with natural annotations: Word segmentation as a case study. In *ACL-2013*, pages 761–769, Sofia, Bulgaria, August 2013. ACL.

[Kong *et al.*, 2015] Lingpeng Kong, Chris Dyer, and Noah A. Smith. Segmental recurrent neural networks. *CoRR*, abs/1511.06018, 2015.

[Lafferty *et al.*, 2001] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML 01*, pages 282–289, San Francisco, CA, USA, 2001.

[Ling *et al.*, 2015] Wang Ling, Chris Dyer, Alan W Black, and Isabel Trancoso. Two/too simple adaptations of word2vec for syntax problems. In *NAACL-2015*, pages 1299–1304, Denver, Colorado, May–June 2015. ACL.

[Mikolov *et al.*, 2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013.

[Okanohara *et al.*, 2006] Daisuke Okanohara, Yusuke Miyao, Yoshimasa Tsuruoka, and Jun'ichi Tsujii. Improving the scalability of semi-markov conditional random fields for named entity recognition. In *ACL-2006*, pages 465–472, Sydney, Australia, July 2006. ACL.

[Passos *et al.*, 2014] Alexandre Passos, Vineet Kumar, and Andrew McCallum. Lexicon infused phrase embeddings for named entity resolution. In *CoNLL-2014*, pages 78–86, Ann Arbor, Michigan, June 2014. ACL.

[Pei *et al.*, 2014] Wenzhe Pei, Tao Ge, and Baobao Chang. Max-margin tensor neural network for chinese word segmentation. In *ACL-2014*, pages 293–303, Baltimore, Maryland, June 2014. ACL.

[Ratinov and Roth, 2009] Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *CoNLL-2009*, pages 147–155, Boulder, Colorado, June 2009. ACL.

[Sarawagi and Cohen, 2004] Sunita Sarawagi and William W. Cohen. Semi-markov conditional random fields for information extraction. In *NIPS 17*, pages 1185–1192. MIT Press, Cambridge, MA, 2004.

[Sun *et al.*, 2009] Xu Sun, Yaozhong Zhang, Takuya Matsuzaki, Yoshimasa Tsuruoka, and Jun'ichi Tsujii. A discriminative latent variable chinese segmenter with hybrid word/character information. In *NAACL-2009*, pages 56–64, Boulder, Colorado, June 2009. ACL.

[Tseng, 2005] Huihsin Tseng. A conditional random field word segmenter. In *In Fourth SIGHAN Workshop on Chinese Language Processing*, 2005.

[Wang *et al.*, 2011] Yiou Wang, Jun'ichi Kazama, Yoshimasa Tsuruoka, Wenliang Chen, Yujie Zhang, and Kentaro Torisawa. Improving chinese word segmentation and pos tagging with semi-supervised methods using large auto-analyzed data. In *IJCNLP-2011*, pages 309–317, Chiang Mai, Thailand, November 2011. AFNLP.

[Yang and Cardie, 2012] Bishan Yang and Claire Cardie. Extracting opinion expressions with semi-markov conditional random fields. In *EMNLP-2012*, pages 1335–1345, Jeju Island, Korea, July 2012. ACL.

[Zhang and Clark, 2007] Yue Zhang and Stephen Clark. Chinese segmentation with a word-based perceptron algorithm. In *ACL-2007*, pages 840–847, Prague, Czech Republic, June 2007. ACL.

[Zheng *et al.*, 2013] Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. Deep learning for Chinese word segmentation and POS tagging. In *EMNLP-2013*, pages 647–657, Seattle, Washington, USA, October 2013. ACL.