Investigating Self-Attention Network for Chinese Word Segmentation

Leilei Gan and Yue Zhang

Abstract—Neural network has become the dominant method for Chinese word segmentation. Most existing models cast the task as sequence labeling, using BiLSTM-CRF for representing the input, and making output predictions. Recently, attention-based sequence models have emerged as a highly competitive alternative to LSTMs, which allow better running speed by parallelization of computation. We investigate self-attention network (SAN) for Chinese word segmentation, making comparisons between BiLSTM-CRF models. In addition, the influence of contextualized character embeddings is investigated using BERT, and a method is proposed for integrating word information into SAN segmentation. Results show that SAN gives highly competitive results compared with BiLSTMs, with BERT, and word information further improving segmentation for in-domain, and cross-domain evaluation. Our final models give the best results for 6 heterogenous domain benchmarks.

Index Terms—Chinese word segmentation, contextualized word embedding, domain adaptation, self-attention network.

I. INTRODUCTION

Word segmentation is a necessary pre-processing step for Chinese natural language processing tasks [3], [10], [25], [30], [38]. The dominant method treats Chinese Word Segmentation (CWS) as a sequence labeling problem [31], and neural network models [2], [32], [36], [39] have achieved the state-of-the-art results. A representative model [4], [5] takes LSTM [9] as a feature extractor, and a standard CRF [13] layer is used on top of BiLSTM layers to predict the label sequences.

As an alternative representation learning model, self-attention network (SAN) [28] has been shown highly effective for a range of natural language processing tasks, such as machine translation [27], constituency parsing [12], semantic role labeling [26], and language modeling [6], [21], [42]. Compared with recurrent neural networks (RNNs) [7], SAN has advantages of capturing long-term dependencies and supporting parallel computing more easily. It has become a dominant approach for learning sequence representation in the research literature in recent years. However, its effectiveness on CWS has not been fully investigated in the literature.

We empirically investigate SAN for CWS by building a SAN-CRF word segmentor, comparing its effectiveness with a state-of-the-art BiLSTM-CRF baseline. In particular, we take the Transformer [28] architecture for building our model. The default Transformer framework performs global attention, where each input character learns its representation by attention over all the other characters in the input sentence. On the other hand, it has been shown that a local attention mechanism has its unique advantages over global attention for certain NLP tasks [43]–[45]. Observing that Chinese word segmentation can rely heavily on local information, we compare global attention and local attention in the model.

Based on the SAN-CRF segmentation model, we investigate two further questions. First, in Chinese, characters are highly polysemantic, where the same character can have different meanings in different contexts. As a result, contextualized character representations, which vary according to sentence context, can be potentially more useful for Chinese word segmentation compared with static embeddings, which most previous work uses [5], [33], [37]. SAN has also been shown to be a useful method for training contextualized word representations [6], [21]. We compare context-independent character representations [17], [18] with contextualized character representations using our SAN framework in both in-domain and cross-domain CWS evaluation.

Second, out-of-vocabulary (OOV) words, especially domain-specific noun entities, raise a challenge for cross-domain CWS. To solve this problem, domain lexicons can be used [35], [37] for cross-domain CWS tasks. We consider a novel method for integrating lexicons to the SAN framework for cross-domain CWS, using attention to integrate word information by generalizing words into POS tags, resulting in end-to-end neural type-supervised domain adaptation.

Results on three benchmarks show that SAN-CRF can achieve competitive performance compared with BiLSTM-CRF. In addition, BERT character embeddings are used for both in-domain and cross-domain evaluation. One important observation is that local attention plays a crucial role in the competitiveness of SAN segmentation model. In cross-domain evaluation, our proposed neural type-supervised method gives an averaged error reduction of 30.32% on three cross-domain datasets. Our method gives the best results on standard benchmarks including CTB, PKU, MSR,
ZX, FR and DL. To the best of our knowledge, we are the first to investigate SAN for CWS.\footnote{Online}.\footnote{There has been recent unpublished work on this subject also \cite{41} which is done contemporarily.}

II. RELATED WORK

Our work is related with three strands of existing work, including: i) methods on neural Chinese word segmentation, ii) recent investigation on self-attention network and iii) the use of contextualized word representations for improving NLP tasks.

A. Chinese Word Segmentation

There has been a long line of work on neural word segmentation. \cite{4}, \cite{5} extract features based on character representation by using LSTM or GRU, while \cite{46} investigate using convolution neural network as a feature extractor. \cite{36} propose a transition-based neural model, which can make use of the word-level features. \cite{39} train character embedding with word-based context information on auto-segmented data. \cite{32} exploit the effectiveness of rich external resources through multi-task learning. \cite{47} jointly train Chinese word segmentation and dependency parsing using a graph-based model. While the above work shows that CNN and LSTM are useful for CWS, we show that SAN with local attention can give strong accuracies. For cross-domain CWS, \cite{35} propose a type-supervised domain adaptation approach for joint CWS and POS-tagging, which shows competitive results compared to token-supervised methods. \cite{20} investigate CWS for Chinese novels, proposing a method to automatically mine noun entities for novels using a double-propagation algorithm. \cite{37} investigate how to integrate external dictionaries into CWS models. Similar to \cite{35} and \cite{37}, our work uses a domain lexicon. To our knowledge, we are the first to investigate contextualized character embeddings and lexicon integration for neural CWS under a SAN framework.

B. Self-Attention Network

Self-attention network \cite{28} was first proposed for machine translation. \cite{26} and \cite{24} use SAN for the task of semantic role labeling, which can directly capture the relationship between two arbitrary tokens in the sequence. \cite{24} incorporate linguistic information through multi-task learning, including dependency parsing, part-of-speech and predicate detection. \cite{22} propose multi-dimensional attention as well as directional information, achieving the state-of-the-art results on natural language inference and sentiment analysis. \cite{12} show that a novel encoder based on self-attention can lead to state-of-the-art results for the constituency parsing task. Along with this strand of work, we study the influence of global and local attention for CWS. We build a SAN-CRF word segmentor, which gives competitive results compared with BiLSTMs. To our knowledge, we are the first to investigate SAN for Chinese word segmentation.\footnote{Example: ELMo \cite{19} is based on recurrent neural network language models. OpenAI GPT \cite{21} builds a left-to-right language model with a multi-layer multi-head self-attention networks, which can handle long-range dependencies better compared to recurrent networks. Different from OpenAI GPT, BERT \cite{6} is trained from large scale raw texts using masked language model task with a deep bidirectional Transformer. \cite{40} propose a multi-criteria method of CWS based on BERT, which uses a private projection layer to learn segmentation criteria of each dataset, and a shared projection layer to learn their common basic knowledge. However, they did not verify their method on cross-domain datasets. Our work investigates the effect of contextualized character representation on both in-domain and cross-domain CWS under a unified SAN framework.}

C. Contextualized Word Representation

Context-dependent word representations pre-trained from large-scale corpora have received much recent attention. For example, ELMo \cite{19} is based on recurrent neural network language models. OpenAI GPT \cite{21} builds a left-to-right language model with a multi-layer multi-head self-attention networks, which can handle long-range dependencies better compared to recurrent networks. Different from OpenAI GPT, BERT \cite{6} is trained from large scale raw texts using masked language model task with a deep bidirectional Transformer. \cite{40} propose a multi-criteria method of CWS based on BERT, which uses a private projection layer to learn segmentation criteria of each dataset, and a shared projection layer to learn their common basic knowledge. However, they did not verify their method on cross-domain datasets. Our work investigates the effect of contextualized character representation on both in-domain and cross-domain CWS under a unified SAN framework.

III. BASELINE

We take BiLSTM-CRF as our baseline, which has been shown giving the state-of-the-art results \cite{5}, \cite{33}. Formally, given an input sentence with \( m \) characters \( s = c_1, c_2, \ldots, c_m \), where \( c_i \) denotes the \( i \)th character, the task of character-based CWS is to assign each character \( c_i \) with a label \( y_i \), where \( y_i \in \{ B, M, E, S \} \) \cite{31}. The labels \( B, M, E \) and \( S \) represent the beginning, middle and end of a word, and single character word, respectively.

For each character \( c_i \), its input representation is the concatenation of unigram character embedding \( e_{c_i} \) and bigram character embedding \( e_{c_{i-1},c_{i}} \), as follows:

\[
   x_i^c = e_{c_i} \oplus e_{c_{i-1}, c_{i+1}},
\]

where \( \oplus \) represents concatenation operation.

Following \cite{5}, we feed the character representations \( x_i^c \) into BiLSTM layers to capture forward and backward hidden states. The final hidden representation of character \( c_i \) is the concatenation of forward hidden state \( \hat{h}_f \) and backward hidden state \( \hat{h}_b \):

\[
   h_i = \hat{h}_f \oplus \hat{h}_b,
\]

On top of the hidden representations, a CRF layer is used to consider the dependencies of adjacent labels. Formally, the probability of a label sequence \( y = y_1, y_2, \ldots, y_n \) of sentence \( s \) is:

\[
   P(y|s) = \frac{\exp \left( \sum_{i=1}^{n} (F(y_i) + L(y_{i-1}, y_i)) \right)}{\sum_{y' \in \mathcal{C}(s)} \exp \left( \sum_{i=1}^{n} (F(y'_i) + L(y'_{i-1}, y'_i)) \right)},
\]

where \( \mathcal{C}(s) \) is the set of all possible label sequences \( s. F(y_i) = W^f_i h_i + b^f_i \) is the emission score from hidden representation \( h_i \) to label \( y_i \). \( W^f_i \) and \( b^f_i \) are label-specific parameters. \( L(y_{i-1}, y_i) \) is the transition score from \( y_{i-1} \) to \( y_i \), which is a scalar model parameter specific to a pair of labels.

IV. MODEL

Fig. 1 shows our segmentor framework on an input character sequence “中国科学院院士 (Fellow of the Chinese Academy of Sciences)”. The model takes character representation and positional embeddings as input. By matching the input to a word-POS lexicon, word information is investigated by using attention for each character. Multiple layers of self-attention...
network [28] are used as feature extractor to replace BiLSTM in the baseline. Similar to the baseline, we also use a CRF layer on top of the self-attention network to model the dependencies of adjacent labels.

A. Embedding Layer

As shown in Fig. 1, the embedding layer consists of character embeddings and positional embeddings. The character representation of $c_i$ is the concatenation of unigram character embedding $e_{c_i}$ and bigram character embedding $e_{c_i c_{i+1}}$:

$$\tilde{x}_i = e_{c_i} \oplus e_{c_i c_{i+1}}.$$  

Because self-attention network does not explicitly consider sequence information, positional encoding is added to the input embeddings $\tilde{x}_i$ of self-attention network as follows:

$$PE_{(pos, 2i)} = \sin(pos/10000^{2i/d_{model}})$$
$$PE_{(pos, 2i+1)} = \cos(pos/10000^{2i/d_{model}})$$

$$x_i = \tilde{x}_i + PE,$$  

where $pos$ is the position, $i$ is the dimension, $d_{model}$ is the dimension of output, respectively, and $+$ denotes vector addition. $x_i^f$ is the final representation of the character $c_i$.

B. Self-Attention Network

We extend the model of [28] for the SAN segmentor, which has multiple identical layers, each being composed of a multi-head self-attention sub-layer and a position-wise fully connected feed-forward network.

Multi-head self-attention is used to exchange information directly between positions in the sequence. First, for single-head self-attention, the hidden representations of the input representation $X$ is computed by scaled dot-product attention as follows:

$$\text{Attention}(X) = \text{Attention}(Q, K, V)$$
$$= \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) V,$$  

where $Q = W_Q^T X, K = W_K^T X, V = W_V^T X$ are query, key and value vectors, respectively, $d_k$ is the dimension of key and value vectors, and $W_Q, W_K, W_V$ are parameters.

Local Self-Attention In order to investigate the effect of long-term dependencies on CWS task, we propose a local self-attention, which only considers the surrounding positions for each character instead of all positions in the sequence. The intuition is that long-term dependencies may bring more noise than useful information for a sequence labeling task [15]. The local self-attention is denoted as:

$$L\text{-Attention}(X) = \text{Attention}(Q, K, V)$$
$$= \left( \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) \odot W \right) V,$$  

where $\odot$ means element-wise product, $W$ is a mask matrix to control the self-attention inner a window and its element $W_{ij}$ is denoted as:

$$W_{ij} = \begin{cases} 
1 & j - i \leq WS \\
-\infty & \text{otherwise}
\end{cases}.$$  

Here $WS$ is the window size.

Multi-Head Self-Attention is used, which linearly maps $Q, K$ and $V$ into multiple versions $Q_i, K_i$ and $V_i$ and then concatenates the outputs of different head, as follows:

$$\text{MH}(X) = \text{Concat}(head_1, \ldots, head_h)W^o,$$  

where

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V).$$
\[ W^o, W^Q, W^K, W^V \] are parameters.

On top of the multi-head attention sub-layer, a fully connected feed-forward network (FFN) is applied to each position. FFN is composed of two linear transformations with a ReLU activation:

\[
\text{FFN}(x) = W_2 \text{ReLU}(0, xW_1 + b_1) + b_2, \tag{11}
\]

where \( W_1, W_2, b_1 \) and \( b_2 \) are parameters.

V. Investigating Rich Character and Word Features for SAN CWS

We incorporate rich character and word features into the SAN model. In particular, pre-trained contextualized character representation is introduced as well as a word-based neural type-supervised domain adaptation method.

A. BERT Character Representation

BERT [6] is trained from a large-scale corpora by using a deep bidirectional Transformer for masked LM tasks. Usages of BERT can be divided into feature-based and fine-tuning methods. The former fixes all model parameters and directly extracts character features from the pre-trained model, while the latter jointly fine-tunes all parameters on downstream tasks. We take the latter method, feeding the input sequence of characters into BERT and use the top layer output as character representation. Development experiments show that fine-tuning BERT embeddings give better results than the feature-based method.

Formally, the unigram character embedding of character \( c_i \) is replaced by using pre-trained BERT embedding according to the whole sentence.

\[ e_{c_i} = e_{\text{bert}}^c(c_i), \tag{12} \]

where \( e_{\text{bert}}^c \) denotes a pre-trained BERT character embedding.

B. Integrating Word-POS Lexicon for Type-Supervision

We integrate word information into SAN to handle rare words in cross-domain settings. Following the definition by [35], we describe this model in a cross-domain setting only, where \( C_s \) denotes a set of annotated source-domain sentences, and \( \xi_t \) denotes an annotated target-domain lexicon, in which each word is associated with one POS tag. The domain adaptation model is first trained on \( C_s \) and makes use of \( \xi_t \) when performing target domain segmentation.

As shown in Fig. 1, for each character \( c_i \) in the input sentence, the set of all character subsequences that match words in the external lexicon \( D \) is denoted as \( w^i = \{ w_{b_1,e_1}, w_{b_2,e_2}, \ldots, w_{b_m,e_m} \} \). Here \( b_k \) and \( e_k \) are the start and end index of the matched words in the sentence, where \( e_k = i \) and \( b_k \leq i \). Word embeddings should intuitively be used for encoding \( w_{b_k,e_k} \). However, for characters forming domain specific words, there may not be readily available embeddings. POS embeddings can be used as alternative unlexicalized features of words embeddings. We introduce how to integrate POS embeddings to enrich word information from both prediction and training.

Prediction During testing, we additionally match character subsequences in a given input sentence to a word-POS lexicon \( \xi_t \). For each matching subspan, we find a vector representation by first performing a lookup action to a word embedding table, and then using the corresponding POS embedding to represent the word if no word embedding is available for the subspan as follows:

\[
x^w_{b_k,e_k} = \begin{cases} e^w(p_k) & \text{if } w_{b_k,e_k} \in D \text{ and } p_k \text{ is the POS tag of } w_{b_k,e_k} \\ p^e(c_k) & \text{otherwise} \end{cases}, \tag{13}
\]

where \( p_k \) is the corresponding POS tag of \( w_{b_k,e_k} \) provided by \( \xi_t \). \( e^w \) and \( p^e \) are word and POS embedding lookup tables, respectively.

Training Training is performed on a source domain corpus only. We do not fine-tune word embeddings. The key task for knowledge transfer is the learning of POS embeddings, which offers a generalized representation for words not in the embedding lexicon. To this end, we randomly replace words in the training data with their gold-standard POS tags as follows:

\[
P(w_{b_k,e_k}) = \min \left( 1, \sqrt{\frac{t}{f(w_{b_k,e_k})}} \right), \tag{14}
\]

where \( f(w_{b_k,e_k}) \) is the frequency of \( w_{b_k,e_k} \) in the training data and \( t \) is a chosen threshold.

The representation of \( w_{b_k,e_k} \) is:

\[
x^w_{b_k,e_k} = \begin{cases} p^e(p_k) & 0 \leq rb < P(w_{b_k,e_k}) \\ e^w(p_k) & \text{otherwise} \end{cases}, \tag{15}
\]

where \( rb \) is a random number and \( p_k \) is the gold-standard POS tag of \( w_{b_k,e_k} \). We name this method as integral POS embedding method (\( P_i \)).

Considering the positional information of characters in the word, the set of POS tags can be denoted in combination with segmentation labels: \( L = \{ p_{lb}, p_{lm}, p_{le}, p_{lbm}, \ldots, p_{lne} \} \). We name this method as individual POS embedding method (\( P_{i,b} \)). The difference between \( P_i \) and \( P_{i,b} \) is that for \( c_j \) and matched word \( w_{b_k,e_k} \), if \( c_j \) is the first, middle or last character of \( w_{b_k,e_k} \), the corresponding POS tag of \( w_{b_k,e_k} \) is \( p_{lb}, p_{lm} \) and \( p_{le} \), respectively.
respectively. Fig. 2 shows the difference between $P_t$ and $P_{c,b}$ through an example.

For each character $c_i$, we integrate dictionary word information by augmenting its embedding with a word context vector $h_i$, which is the weighted sum over $x_{b_k,c_k}$ for all spans $(b_k,c_k)$ that contain $c_i$. In particular,

$$h_i = \sum \alpha_{ik} x_{b_k,c_k}^w,$$  \hspace{1cm} (16)

where the weight for each context word is:

$$\alpha_{ik} = \text{attention}(x_i^c, x_{b_k,c_k}^w) = \frac{\exp(\text{score}(x_i^c, x_{b_k,c_k}^w))}{\sum_{k=1}^m \exp(\text{score}(x_i^c, x_{b_k,c_k}^w))},$$  \hspace{1cm} (17)

Considering computation efficiency, the score function is:

$$\text{score}(x_i^c, x_{b_k,c_k}^w) = x_i^c W x_{b_k,c_k}^w;$$  \hspace{1cm} (18)

where $W$ denotes a model parameter. The output of the attention layer is the concatenation of the character embedding $x_i^c$ and the context vector $h_i$:

$$x_i = x_i^c \oplus h_i.$$

C. Decoding and Training

For decoding, the Viterbi algorithm [29] is used to find the highest scored label sequence $y^*$ over a input sentence.

For training a set with $N$ gold-standard samples, the loss function is negative log-likelihood of sentence-level with $L_2$ regularization:

$$\text{Loss} = - \sum_{i=1}^N \log(P(y_i|s_i)) + \frac{\lambda}{2}||\Theta||^2_2;$$  \hspace{1cm} (20)

where $\lambda$ and $\Theta$ represent $L_2$ regularization parameter and the model parameter set, respectively.

VI. EXPERIMENTS

We carry out an extensive set of experiments to investigate the effectiveness of SAN-CRF and the proposed neural type-supervised domain adaptation method across different domains under different settings. F1-value is taken as our main metric.

A. Datasets

We separately evaluate the proposed model in in-domain and cross-domain settings. For in-domain evaluation, CTB6 (Chinese Tree Bank 6.0), PKU and MSR are taken as the datasets. The train/dev/test split of CTB6 follows [36], while the split of PKU and MSR are taken from the SIGHAN Bakeoff 2005 [8]. For cross-domain evaluation, PKU and MSR are taken from the SIGHAN Bakeoff 2005 [8]. The train/dev/test split of CTB6 follows [36], while the split of the target domain. Following [35], we collect target-domain lexicons from Internet Encyclopedia$^3$.$^4$.$^5$ and annotate every word in the lexicons with one POS tag. Table I shows the statistics of the datasets.

B. Experimental Settings

Training Details Table II shows the values of model hyper-parameters. For the SAN CWS model, we use the Adam [11] optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.98$, $\epsilon = 10^{-9}$. Following [28], we increase the learning rate linearly for the first warmup_steps steps, and then decrease it proportionally. The value of warmup_steps is set to 1000. When BERT is used for character embeddings, the learning rate is set to 5e-6. For the baseline model, we use stochastic gradient descent (SGD) following [33], and the initial learning rate is set to 0.001, which gives better development results.

Baseline Settings For the BiLSTM baseline, we ran a grid search on the CTB6 dataset to find the best values of BiLSTM layer and hidden dimension. The results are listed in Table III, which shows that the model with 3 layers and 100 hidden dimension achieves the best development result. We use this setting as our BiLSTM baseline model.

---

$3[^{Online}]$: Available: https://baike.baidu.com/item/互联网/12418
$4[^{Online}]$: Available: https://baike.baidu.com/item/互联网/64139
$5[^{Online}]$: Available: https://baike.baidu.com/item/互联网/5313

---

TABLE I

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PKU</td>
<td></td>
</tr>
</tbody>
</table>

TABLE II

<table>
<thead>
<tr>
<th>Hyper-Parameter Values</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Char emb size</td>
<td></td>
</tr>
<tr>
<td>Word emb size</td>
<td></td>
</tr>
<tr>
<td>Bigram emb size</td>
<td></td>
</tr>
<tr>
<td>BERT emb size</td>
<td></td>
</tr>
<tr>
<td>LSTM layer</td>
<td></td>
</tr>
<tr>
<td>LSTM hidden</td>
<td></td>
</tr>
<tr>
<td>LSTM input dropout</td>
<td></td>
</tr>
<tr>
<td>Batch size</td>
<td></td>
</tr>
</tbody>
</table>

TABLE III

<table>
<thead>
<tr>
<th>Effect of Numbers of BiLSTM Layer and Hidden Dimension on the CTB6 Development Dataset</th>
<th>F1-Value</th>
<th>Model Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Layer</td>
<td>Hidden Dimension</td>
<td>F1-Value</td>
</tr>
<tr>
<td>--------</td>
<td>------------------</td>
<td>---------</td>
</tr>
<tr>
<td>1</td>
<td>100</td>
<td>95.48</td>
</tr>
<tr>
<td>1</td>
<td>200</td>
<td>95.54</td>
</tr>
<tr>
<td>1</td>
<td>300</td>
<td>95.42</td>
</tr>
<tr>
<td>1</td>
<td>400</td>
<td>95.34</td>
</tr>
<tr>
<td>1</td>
<td>500</td>
<td>95.38</td>
</tr>
<tr>
<td>1</td>
<td>600</td>
<td>95.56</td>
</tr>
<tr>
<td>1</td>
<td>700</td>
<td>95.46</td>
</tr>
<tr>
<td>1</td>
<td>800</td>
<td>95.44</td>
</tr>
<tr>
<td>1</td>
<td>900</td>
<td>95.38</td>
</tr>
<tr>
<td>1</td>
<td>1000</td>
<td>95.40</td>
</tr>
<tr>
<td>1</td>
<td>1100</td>
<td>95.34</td>
</tr>
</tbody>
</table>
Character and Word Embedding The pre-trained word embedding size is 200, which is based on word co-occurrence and the directions of word pairs [23], and the word length is restricted to 4. We use the topmost layer output as character embedding of the pre-trained Chinese Simplified BERT model with 12 layers, 768 hidden units and 12 heads.\(^6\) Besides that, the bigram embeddings and character unigram embeddings used for attending words are pre-trained using word2vec [1], which is the same as following [36].

C. Development Experiments

We perform development experiments on the CTB6 development dataset to investigate the influence of hyper-parameters of self-attention network for CWS, and compare the performance of SAN, especially local self-attention, with BiLSTM. In addition, we evaluate the effect of utilizing of BERT for CWS models. We denote the original self-attention network as “SAN,” and local self-attention network as “L-SAN”. “BiLSTM” is our baseline model, which uses a bidirectional LSTM as feature extractor.

Effect of Local Attention As Fig. 3 shows, the performance of “L-SAN+CRF” models with different window sizes give better results compared to “SAN+CRF,” which suggests that long-range global context can bring more noise and harm segmentation. In addition, the proposed local self-attention network model achieves competitive results compared with the baseline BiLSTM model. Fig. 3 also shows the impact of different window sizes on the performance. First, we find that “L-SAN” with window size 5 achieves the best result. Second, when we increase the window size larger than 5, the performance decreases. We hypothesize that a larger window size brings more noise than information. However, if we set the window size smaller than 5, the performance also decreases, which is likely because a small window size does not capture sufficient context. Thus, we choose 5 as the window size for the local self-attention network.

Effect of BERT As shown in Fig. 4, by replacing word2vec character embeddings with BERT, both BiLSTM and L-SAN models reach the best F1-value within several epochs, with a significant improvement. We further demonstrates the effect of BERT by replacing the CRF layer with a softmax classification layer, which still achieves strong performance, proving that context-dependent word representation can benefit CWS even without a CRF layer.

Width and Depth [28] show that increasing the model size can improve the performance of English-to-German translation. We investigate the effect of number of layers and heads on CWS by varying the number of layers between 2 and 6, and varying the number of heads from 2 to 8. The dimension of head is fixed to 50. The results and total numbers of parameters are listed in Table IV. We found that the model can achieve the best F1-value 96.0 with only 2 layers and 2 heads, yet the F1-value does not increase simultaneously with the increase of model size. According to this result, we fix the number of layers and heads to 2 and 2 for the remaining experiments, respectively.

D. Final Results

In-Domain Results We evaluate our model on three news datasets, including CTB, PKU and MSR. The main results and the results of recent state-of-the-art models are listed in Table V. Compared with the baseline “BiLSTM+CRF” model, the proposed “L-SAN+CRF” model achieves similar results, which

\(^6\)https://github.com/huggingface/pytorch-pretrained-BERT
proves that self-attention network can be a competitive feature extractor for CWS. When replacing word2vec character embedding with BERT, the “BiLSTM+BERT+CRF” model gives 46.8%, 34.0% and 37.0% error reduction on CTB6/PKU/MSR, respectively, and the “L-SAN+BERT+CRF” model has 42.2%, 32.7% and 44.8% error reductions on three in-domain datasets, respectively. Fig. 5 also shows that replacing the CRF layer with a softmax classification layer can still achieve competitive results, which demonstrates the strong effect of BERT. We also try to remove BiLSTM or L-SAN module and directly inference based on the output of BERT. The results again show the strong effect of BERT on in-domain CWS.

Cross-Domain Results

We evaluate our model on the three cross-domain datasets, including ZX, FR and DL. The main results and results of three state-of-the-art models are listed in Table VI. “t” means that a neural type-supervised method is used to learn POS embeddings and domain-specific words are generalized to corresponding tags. In “t\_b”, we learn different POS embeddings for different positions in a word, as mentioned in Section V. As shown in Table VI, the F1-values of “L-SAN+BERT+CRFT” have an average 0.7 improvement compared with the state-of-the-art results [34] on ZX and FR without using [34]’s domain adaptation techniques, which shows that contextualized BERT representation can stabilize the performance against the sentence length. However, the two models using BERT show more stable performance-length curves, which shows that contextualized BERT representation can stabilize the performance against the sentence length.

E. Analysis

Sentence Length

As shown in Fig. 5, we compare the baseline model and local self-attention network model, as well as the two models with BERT input representation against different sentence lengths on the CTB6 test dataset. The two models without using BERT show a similar performance-length curves, which reach a peak at around 30-character sentences and decrease when sentence length over 90. One possible reason is that very short sentences are rare while long sentences are semantically more challenging. However, the two models using BERT show more stable performance-length curves, which shows that contextualized BERT representation can stabilize the performance against the sentence length.

Comparison of Training and Test Speed

We compare the training and test speeds of the L-SAN model with the BiLSTM baseline on the CTB6 dataset. Both models are evaluated on the same server with a GeForce GTX 1080Ti GPU. The batch size is set to 32 for both training and test. The results are listed in Table VII. We can find that the “L-SAN+CRF” model outperforms “BiLSTM+CRF” in both training and test speeds with almost the same model size, which demonstrates the advantage of parallelization by SAN.
Noun Entity Segmentation  Noun entities raise a key problem for cross-domain CWS. Table IX shows segmentation results of the three models on the 15 most frequent noun entities on the three datasets. M1 and M2 represent “L-SAN+CRF+BERT” and “L-SAN+CRF+BERT+rt,” respectively, while M3 represents “L-SAN+CRF+BERT+rt+b.” As the table shows, the average precision of M1 is 0.55. By using neural type-supervised domain adaptation method, the average precision of M2 has an improvement of 0.18 in absolute value. Some person names are incorrectly segmented by M2, such as “戴沐白(Person Name)” and “叶知秋(Person Name)”.

VII. CONCLUSION

We investigated self-attention network (SAN) for Chinese word segmentation, demonstrating that it can achieve competitive results with recurrent network methods. We found that local attention gives better results compared to standard SAN. Under SAN, we also investigate the influence of rich character and word features, including BERT character embeddings and a neural attention method to integrate word information into character-based CWS. Extensive in-domain and cross-domain experiments show that the proposed SAN method achieves the state-of-the-art performance on both in-domain and cross-domain Chinese word segmentation datasets.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their insightful comments.

REFERENCES


Leilei Gan received the B.S. degree from the School of Computer Science and Technology, Nanjing University of Posts and Telecommunications, Nanjing, China, and the M.S. degree from the Department of Computer Science and Technology, Nanjing University, Nanjing, China. He is currently working toward the Ph.D. degree with the School of Computer Science and Technology, Zhejiang University, Hangzhou, China. His research interest is mainly focused on natural language processing.

Yue Zhang is currently an Associate Professor with Westlake University, Hangzhou, China. His research interests include natural language processing and computational finance. He has been working on statistical parsing, text synthesis, natural language synthesis, machine translation, information extraction, sentiment analysis, and stock market analysis intensively. He won the best paper awards of IALP 2017 and COLING 2018. He serves the editorial board for the Transactions of the Association of Computational Linguistics (Action Editor), ACM Transactions on Asian and Low-Resource Language Information Processing (Associate Editor), and IEEE TRANSACTIONS ON BIG DATA (Associate Editor), and as area chairs of COLING 2014/18, NAACL 2015/19, EMNLP 2015/17/19, and ACL 2017/18/19.