Lexicon-Based Graph Convolutional Network for Chinese Word Segmentation

Kaiyu Huang  Hao Yu  Junpeng Liu
Wei Liu  Jingxiang Cao  Degen Huang∗
School of Computer Science, Dalian University of Technology
{kaiyuhuang, yuhaodlut, liujunpeng_nlp, liuweidlut}@mail.dlut.edu.cn
{caojx, huangdg}@dlut.edu.cn

Abstract

Precise information of word boundary can alleviate the problem of lexical ambiguity to improve the performance of natural language processing (NLP) tasks. Thus, Chinese word segmentation (CWS) is a fundamental task in NLP. Due to the development of pre-trained language models (PLM), pre-trained knowledge can help neural methods solve the main problems of the CWS in significant measure. Existing methods have already achieved high performance on several benchmarks (e.g., Bakeoff-2005). However, recent outstanding studies are limited by the small-scale annotated corpus. To further improve the performance of CWS methods based on fine-tuning the PLMs, we propose a novel neural framework, LbgGCN, which incorporates a lexicon-based graph convolutional network into the Transformer encoder. Experimental results on five benchmarks and four cross-domain datasets show the LbgGCN successfully captures the information of candidate words and helps to improve performance on the benchmarks (Bakeoff-2005 and CTB6) and the cross-domain datasets (SIGHAN-2010). Further experiments and analyses demonstrate that our proposed framework effectively models the lexicon to enhance the ability of basic neural frameworks and strengthens the robustness in the cross-domain scenario.1

1 Introduction

Neural methods often leverage word-level information to improve the performance of many downstream natural language processing (NLP) tasks such as text classification and machine translation (Yang et al., 2018), etc. Therefore, in determining the word boundary, word segmentation is regarded as a prerequisite for most downstream NLP tasks.

∗Corresponding author
1Source codes of this paper are available on https://github.com/koukaiu/lbgcn

Unlike most written languages, the Chinese written language has no explicit delimiters to separate words in the written text. Thus, Chinese word segmentation (CWS) is an essential and pre-processing step for many Chinese NLP tasks.

With the development of deep learning techniques, recent neural CWS approaches that do not heavily rely on the hand-craft feature engineering have already achieved high performance on several benchmark datasets (Cai and Zhao, 2016; Cai et al., 2017; Ma et al., 2018). In particular, recent outstanding studies have also exploited the learning paradigm in applying pre-trained language models (PLM) for many NLP tasks. Various methods that fine-tune PLMs have achieved progress on in-domain and cross-domain CWS without much manual effort (Meng et al., 2019; Huang et al., 2020; Tian et al., 2020; Ke et al., 2021).

Prior research has shown that the problems of CWS are segmentation ambiguity and out-of-vocabulary (OOV) words (Zhao et al., 2019). With the help of the pre-trained knowledge (Devlin et al., 2018; Liu et al., 2019), the fine-tuning CWS methods can effectively alleviate these two issues and outperform other neural network architectures. The methods fine-tuning PLMs become the mainstream approach for CWS. However, the performance of fine-tuning CWS methods is limited by the scale and quality of annotated CWS corpus. The dependencies between neighboring Chinese characters are diverse and it is hard to build a large-scale annotated corpus because of the characteristics of linguistics in Chinese. The difficulty of manual annotation restricts the scale and quality of CWS datasets. Besides, directly fine-tuning methods do not utilize contextual n-grams or other contextual information, which is important for previous model architectures (e.g., BiLSTM and Transformer) (Huang et al., 2015; Ma et al., 2018; Qiu et al., 2020). The methods that fine-tune PLMs may generate segmentation errors because of ambigu-
ous contextual information. Thus, it is a challenge
to design a framework that can effectively transfer
pre-trained knowledge into the CWS.

In this paper, we propose the LbGCN, a neural
framework with a lexicon-based graph convolu-
tional network (GCN), to improve the performance
of the CWS by leveraging lexicon knowledge. In
detail, we utilize the GCN to extract contextual
features of candidate words and the information
of word boundary from the pre-defined lexicon.
The neural framework incorporates the GCN into
the Transformer encoder (Vaswani et al., 2017)
which is a part of PLM (e.g., BERT (Devlin et al.,
2018)). The additional lexicon-based GCN can
supply a gap of fine-tuning paradigm and better
transfer pre-trained knowledge into the in-domain
and cross-domain CWS tasks. Besides, through
multi-feature interaction, the disambiguation and
OOV word recognition are effectively carried out.

To sum up, the contributions of this work are as
follows:

• Our proposed framework mainly consists of
a lexicon-based GCN and the Transformer
encoder. The lexicon-based GCN captures
rich contextual information to alleviate the
problem of lack-training by the small-scale
annotated corpus. This framework achieves a
noticeable improvement for CWS.

• Experimental results obtained from widely
used benchmark datasets demonstrate that
LbGCN can improve the performance com-
pared with powerful baseline methods and out-
perform previous state-of-the-art studies.

• The novel method extracts the information
from the lexicon via the GCN and is not over-
reliant on the quality of the lexicon. Exper-
imental results in the cross-domain scenario
prove that the method can enhance the robust-
ness of the basic neural CWS approaches.

2 Related Work

formalizes the CWS as a sequence labeling prob-
lem, most studies follow the character-based
paradigm to predict segmentation labels for each
character in the sentence. In particular, the adopted
methods fall into two categories, including 1) statis-
tical machine learning methods (Peng et al., 2004;
Tseng et al., 2005; Zhao and Kit, 2008; Zhao et al.,
2010) and 2) neural network methods (Zheng et al.,
2013; Pei et al., 2014; Chen et al., 2015a,b; Cai
and Zhao, 2016; Yang et al., 2017). As the studies
of deep learning techniques develop in-depth, the
neural CWS methods achieve better performance
compared with statistical learning methods (Cai
et al., 2017; Zhou et al., 2017; Ma et al., 2018;
Yang et al., 2019a; Wang et al., 2019). And neural
network architectures gradually replace statistical
machine learning methods as the mainstream ap-
proaches for CWS.

Cross-Domain CWS However, there is an obvi-
ous gap in the cross-domain CWS scenario. Neural
CWS methods still suffer from the OOV problems.
To alleviate this problem, many kinds of research
utilize external resources (e.g., pre-trained embed-
ings, unlabeled data, and lexicons) to improve the
performance of the cross-domain CWS (Zhao et al.,
2018; Zhang et al., 2018; Ye et al., 2019; Ding
et al., 2020). For example, Huang et al. (2020) try
to transfer pre-trained knowledge into the cross-
domain CWS in full by leveraging more annotated
datasets with different segmentation criteria (Chen
et al., 2017). Tian et al. (2020) utilize lexicons and
wordhood measures to enhance the robustness in
the cross-domain CWS scenario.

Graph Neural Network In recent years, the
graph neural network has been fully explored and
achieved significant progress in several kinds of
NLP tasks (Zhou et al., 2020). When dealing with
text scenarios, graphs can extract the features from
non-structural data by modeling a set of objects
(nodes) and their relationships (edges). In par-
ticular, we can consider each variable in the text
as a node and the dependencies as edges for the
sequence labeling task. Marcheggiani and Titov
(2017) present a syntactic GCN to solve the prob-
lem of semantic role labeling. Ding et al. (2019)
utilize a multi-graph structure to capture the informa-
tion that the gazetteers offer. In addition, the graph
neural network based on the domain lexicon
is used to learn the local composition features for
medical domain CWS (Du et al., 2020).

3 Proposed Framework

The framework of LbGCN is illustrated in Figure 1.
It mainly consists of two parts: an encoder-decoder
layer and a GCN. In the first part, we utilize the
Transformer as the encoder and the Dense as the de-
coder. The Transformer encoder adopts the PLMs
(e.g., BERT and RoBERTa) which contain rich pre-trained knowledge to train. The pre-trained knowledge can effectively help the model alleviate the problem of OOV word recognition. In the second part, a GCN based on the pre-defined lexicon is built. The generated graph embeddings can make up the deficiency of contextual information from candidate words. In addition, the proposed framework that is integrated with bi-gram features and multiple contextual features can improve the performance of CWS.

Following previous studies (Xue, 2003), we regard the CWS as the character-based sequence labeling task. The framework predicts a tag that represents the position in a word for each character (e.g., tag “B” represents the first character in a word). The process of LbGCN to find the most possible path $\hat{Y}$ can be formalized as:

$$\hat{Y} = \arg\max_{\gamma \in \mathcal{T}^N} p(\gamma | \mathcal{X})$$

where $\mathcal{T}$ denotes the set of all types of segmentation labels, and $N$ is the length of the input sentence $\mathcal{X}$.

The rest of this section describes the architecture of the encoder-decoder layer, the construction of the lexicon graph, and how it is integrated with the GCN, respectively.

3.1 Encoder and Decoder

**Transformer Encoder** Recently, there are several PLMs (e.g., BERT and RoBERTa) that have shown state-of-the-art performance of many NLP tasks. In particular, a modified method based on RoBERTa model is built for the Chinese NLP tasks (Cui et al., 2019). With the previous success on PLMs, we adopt the main architecture (Transformer) as the encoder of our proposed framework, which can straightforwardly leverage the pre-trained knowledge from PLMs for the Transformer encoder because of the similar structure.

The PLM is trained for predicting the word in general. To transfer the pre-trained knowledge into the CWS, we need to fine-tune the PLM by the
annotated corpus of CWS. Given an input sentence $X = x_1 \ldots x_{i-1} x_i \ldots x_n$ from the training data, the input sentence is converted to the corresponding vector embeddings $H = [h_1 \ldots h_{i-1} h_i \ldots h_n]$ in the “Lookup Table” layer, where $H \in \mathbb{R}^{N \times d_{model}}$, $N$ and $d_{model}$ represent the length and the same dimensions with the PLM. To be consistent with the pre-trained process, two tags (“[CLS]” and “[SEP]”) are added to the beginning and the end of each sentence, respectively.

Given an input vector sentence $H \in \mathbb{R}^{N \times d_{model}}$, the Transformer encoder utilizes self-attention layers to extract the contextual feature for each character. The self-attention layer adopts “Scaled Dot-Product Attention” to compute representation.

$$Q, K, V = HW^Q, HW^K, HW^V$$  \hspace{1cm} (2)

$$\text{Attn}(Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) V$$  \hspace{1cm} (3)

where $Q, K, V$ represents a query and a set of key-value pairs through a linear transformation respectively, the matrices $W^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W^K \in \mathbb{R}^{d_{model} \times d_k}$, $W^V \in \mathbb{R}^{d_{model} \times d_v}$ are trainable parameters, and $d_k$ is the dimension of $K$.

Instead of performing a single-head attention function, the Transformer encoder uses the multi-head self-attention layer in order to extract contextual features from different representation spaces and utilizes feed forward network (FFN) to enhance representation ability. Assuming the input of the multi-head self-attention layer is $H$, the output $\hat{H}$ is calculated by

$$Z = \text{LN} \left( H + \text{MultiHead}(H) \right) \hspace{1cm} (4)$$

$$\hat{H} = \text{LN} \left( Z + \text{FFN}(Z) \right) \hspace{1cm} (5)$$

where “LN” indicates the layer normalization (Ba et al., 2016).

**Dense Decoder** A dense layer with $W^D \in \mathbb{R}^{d_{model} \times T_n}$ converts hidden dimensions to the 4-tag set $T = \{B, M, E, S\}$, where $T_n$ presents the size of the tag sets ($T_n = 4$). After linear mapping, the framework adopts the function $\text{Softmax}$ and the greedy search for decoding. In previous studies, many kinds of research adopt the CRF as the decoder layer to improve the performance of sequence labeling tasks (Lample et al., 2016). However, the CRF layer has larger time complexity and space complexity for CWS (Duan and Zhao, 2020). For practicality, the proposed framework utilizes the lightweight function $\text{Softmax}$ as the decoder layer and also achieves competitive performance compared with other studies using the CRF.

$$p(x) = \text{Softmax}(\hat{H} \cdot W^D + b) \hspace{1cm} (6)$$

The training step of the framework is to minimize the errors by solving the following optimization function:

$$\min_{\Theta^1, \Theta^2} J_{\text{seg}}(y(x)|p(x; \Theta^1, \Theta^2))$$  \hspace{1cm} (7)

where $y(x)$ denotes the true labels on the annotated corpus, $\Theta^1$ and $\Theta^2$ are all trainable parameters in the transformer layer and GCN, respectively, and the loss function $J_{\text{seg}}$ is given by:

$$J_{\text{seg}}(y(x)|p(x)) = - \sum_x y(x) \log p(x)$$  \hspace{1cm} (8)

### 3.2 Lexicon-Based GCN

**Lexicon-Based Construction** The bottom part of the Figure 1 starts with a lexicon and we construct the graph by the pre-defined lexicon. Given the input sentence $X = x_1 \ldots x_{i-1} x_i \ldots x_n$, the graph utilizes a pre-defined lexicon to extract candidate words in the sentence after the Transformer encoder. For example, $X = ["水仙花是草本植物"]$ (Daffodils are herbaceous plant) consists of 8 Chinese characters, and the word list $\mathcal{L} = ["水仙花"]$ (daffodils), “是”(are), “草本”(herbaceous), “植物”(plant) is obtained from the lexicon. The lexicon-based graph is defined as $G : = (V, E)$, where $V$ and $E$ are the sets of nodes and edges, respectively. Each character is represented as a node in the graph and adjacent nodes connect to each other by undirected edges for capturing the contextual information. The set of these undirected edges is $E_c$. Besides, we integrate three additional nodes $V_d = (V_B, V_M, V_E)$ with the character set of nodes $V_c$, and the entire set of nodes is $V = V_c \cup V_d$. To extract the information of the word boundary, we also build edges between candidate words $w_i = c_1 \ldots c_n$, $w_i \in \mathcal{L}$ and additional nodes $V_d$. The entire set of edges is $E = E_c \cup E_d$, where $E_d$ represents the set of edges between candidate words and additional nodes. The 1st character $c_1$ in the candidate word connects to the node $V_B$ and $V_M$, and the last character $c_n$ connects to the node $V_M$ and $V_E$. For instance, the candidate word “水仙花” (daffodils) consists of three characters “水”(water), "仙"(fairy) and “花”(flower). In particular, the character node “水” (water) connects to the
node $V_B$ and $V_M$. The character node “flower” (fairy) only connects to the node $V_M$. The character node “flower” (flower) connects to the node $V_M$ and $V_E$. The construction of the lexicon graph is illustrated in Figure 1.

**GCN** After the construction of the lexicon-based graph, we utilize a GCN (Kipf and Welling, 2016) to encode the graph $G$.

$$\tilde{A} = A + I_N, \tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$$

$$\tilde{H}^{(l+1)} = \sigma \left( \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} \tilde{H}^{(l)} \tilde{W}^{(l)} \right) \tag{9}$$

Here, $A$ is the adjacency matrix of the undirected graph $G$. $I_N$ is the identity matrix and $W^{(l)}$ is a layer-specific trainable weight matrix. $\sigma(\cdot)$ denotes the ReLU activation function. $H^{(l)} \in \mathbb{R}^{N \times D}$ is the matrix of hidden states in the $l_{th}$ layer; $H(0) = X$. GCN mainly consists of two matrices. One is the symmetric normalized Laplacian matrix $\tilde{A}$. The other is the layer-specific trainable weight matrix $W^{(l)}$. The GCN can extract the features from the lexicon-based graph. In addition, the weight matrix adopts the random initialization and the learning rate of this layer is different from the transformer encoder.

4 Experiments

4.1 Datasets and Settings

To verify the improvement of our proposed framework LbGCN, we do comparative experiments on both benchmarks (Bakeoff-2005 (Emerson, 2005) and CTB6) and cross-domain datasets (SIGHAN-2010) (Zhao and Liu, 2010). The size of the benchmark is shown in Table 1. We randomly pick 10% sentences from the training data as the development data for tuning hyper-parameters. For the experiments on the cross-domain datasets, we follow the settings of the “PKU” dataset. For consistency, we pre-process the unsegmented sentences, which is similar to the previous paper (Cai et al., 2017). The evaluation values for CWS are F-score and $R_{oov}$.

We utilize three mainstream PLMs for training the Transformer encoder, including XLNET-BASE (Yang et al., 2019b; Cui et al., 2020), BERT-BASE and ROBERTA-WWM (Cui et al., 2019).² To fine-tune PLMs, we tune a few crucial hyper-parameters with the development sets for the model. The hyper-parameters and search ranges are shown in Table 2. We deploy the model on the same device (GPU environment: Nvidia Tesla V100).

4.2 Experimental Results

This section first reports the results of LbGCN with different configurations on five benchmarks and comparison with existing models. Then it describes the effect of LbGCN in the cross-domain scenario.

**Results on Benchmarks** In the benchmark scenario, we verify the validity on Bakeoff-2005 and CTB6 by comparing LbGCN with three different PLMs, i.e., XLNET, BERT, and RoBERTa. As

---

Table 1: The size of the benchmark, the top blocks indicate the CWS benchmarks (Bakeoff-2005 and CTB6) and the bottom blocks indicate the cross-domain CWS datasets (SIGHAN-2010). Note that the cross-domain datasets do not contain the training sample, so we use the “PKU” which is the most similar to them as the training data.

<table>
<thead>
<tr>
<th>Benchmarks</th>
<th>MSR Train</th>
<th>MSR Test</th>
<th>PKU Train</th>
<th>PKU Test</th>
<th>AS Train</th>
<th>AS Test</th>
<th>CITYU Train</th>
<th>CITYU Test</th>
<th>CTB6 Train</th>
<th>CTB6 Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAR #</td>
<td>4,050K</td>
<td>184K</td>
<td>1,826K</td>
<td>173K</td>
<td>5,500K</td>
<td>123K</td>
<td>2,403K</td>
<td>68K</td>
<td>2,368K</td>
<td>107K</td>
</tr>
<tr>
<td>WORD #</td>
<td>2,368K</td>
<td>107K</td>
<td>1,110K</td>
<td>104K</td>
<td>1,826K</td>
<td>173K</td>
<td>1,456K</td>
<td>41K</td>
<td>701K</td>
<td>82K</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cross-domain</th>
<th>LITERATURE</th>
<th>COMPUTER</th>
<th>MEDICINE</th>
<th>FINANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAR #</td>
<td>50K</td>
<td>54K</td>
<td>51K</td>
<td>53K</td>
</tr>
<tr>
<td>WORD #</td>
<td>35K</td>
<td>35K</td>
<td>31K</td>
<td>33K</td>
</tr>
</tbody>
</table>

Table 2: The crucial hyper-parameters and search ranges.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Hidden states</td>
<td>768</td>
</tr>
<tr>
<td>GCN hidden states</td>
<td>[128,256,768]</td>
</tr>
<tr>
<td>Bi-gram embeds</td>
<td>128</td>
</tr>
<tr>
<td>Learning rate</td>
<td>[2e-4,4e-4,2e-5]</td>
</tr>
<tr>
<td>GCN learning rate</td>
<td>[1e-3,4e-4,2e-5]</td>
</tr>
<tr>
<td>Batch size</td>
<td>[64,128,256]</td>
</tr>
<tr>
<td>Dropout</td>
<td>[0.1,0.2,0.4]</td>
</tr>
<tr>
<td>GCN dropout</td>
<td>[0.1,0.2,0.4]</td>
</tr>
<tr>
<td>Hidden layers</td>
<td>12</td>
</tr>
<tr>
<td>Epochs</td>
<td>20</td>
</tr>
</tbody>
</table>

²The PLMs are available at [https://huggingface.co/models](https://huggingface.co/models).
Table 3: Results and comparison with existing models on the benchmarks Bakeoff-2005 and CTB6, LbGCN is trained based on different PLMs and components. The best values are bolded for each column.

<table>
<thead>
<tr>
<th></th>
<th>PKU</th>
<th>MSR</th>
<th>AS</th>
<th>CityU</th>
<th>CTB6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>F</td>
<td>Rs</td>
<td>F</td>
<td>Rs</td>
<td>F</td>
</tr>
<tr>
<td>MA ET AL. (2018)</td>
<td>96.1</td>
<td>78.8</td>
<td>98.1</td>
<td>80.0</td>
<td>96.2</td>
</tr>
<tr>
<td>GONG ET AL. (2019)</td>
<td>96.15</td>
<td>69.88</td>
<td>97.78</td>
<td>64.20</td>
<td>95.22</td>
</tr>
<tr>
<td>MENG ET AL. (2019)</td>
<td>96.7</td>
<td>-</td>
<td>98.3</td>
<td>-</td>
<td>96.7</td>
</tr>
<tr>
<td>QIU ET AL. (2020)</td>
<td>96.41</td>
<td>78.91</td>
<td>98.05</td>
<td>78.92</td>
<td>96.44</td>
</tr>
<tr>
<td>HUANG ET AL. (2020)</td>
<td>96.85</td>
<td>82.35</td>
<td>98.29</td>
<td>81.75</td>
<td>-</td>
</tr>
<tr>
<td>TIAN ET AL. (2020)</td>
<td>96.53</td>
<td>85.36</td>
<td>98.40</td>
<td>84.87</td>
<td>96.62</td>
</tr>
</tbody>
</table>

Table 4: Results and comparison with existing models on the cross-domain datasets SIGHAN-2010, where “LIT., COM., MED., and FIN.” represent the domain of literature, computer, medicine, and finance, respectively. The best values are bolded for each column.

<table>
<thead>
<tr>
<th></th>
<th>LIT.</th>
<th>COM.</th>
<th>MED.</th>
<th>FIN.</th>
<th>AVG.</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIU ET AL. (2014)</td>
<td>92.49</td>
<td>94.07</td>
<td>92.63</td>
<td>95.54</td>
<td>93.68</td>
</tr>
<tr>
<td>CHEN ET AL. (2015b)</td>
<td>92.89</td>
<td>93.71</td>
<td>92.16</td>
<td>95.20</td>
<td>93.49</td>
</tr>
<tr>
<td>CAI ET AL. (2017)</td>
<td>92.90</td>
<td>94.04</td>
<td>92.10</td>
<td>95.38</td>
<td>93.61</td>
</tr>
<tr>
<td>HUANG ET AL. (2017)</td>
<td>94.33</td>
<td>93.99</td>
<td>92.26</td>
<td>95.81</td>
<td>94.10</td>
</tr>
<tr>
<td>ZHAO ET AL. (2018)</td>
<td>93.23</td>
<td>95.32</td>
<td>93.73</td>
<td>95.84</td>
<td>94.53</td>
</tr>
<tr>
<td>ZHANG ET AL. (2018)</td>
<td>94.76</td>
<td>94.70</td>
<td>94.18</td>
<td>96.06</td>
<td>94.93</td>
</tr>
<tr>
<td>HUANG ET AL. (2020)</td>
<td>96.13</td>
<td>96.08</td>
<td>95.21</td>
<td>96.82</td>
<td>96.06</td>
</tr>
</tbody>
</table>

Table 5: Ablation experiments. The baseline (ID:1) is based on the RoBERTa model.

<table>
<thead>
<tr>
<th></th>
<th>ID</th>
<th>Bi-gram</th>
<th>GCN</th>
<th>PKU</th>
<th>MSR</th>
<th>F</th>
<th>Rs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>×</td>
<td>×</td>
<td>97.00</td>
<td>89.80</td>
<td>98.42</td>
<td>84.59</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>×</td>
<td>-0.02</td>
<td>+0.17</td>
<td>+0.01</td>
<td>+0.42</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>√</td>
<td>+0.15</td>
<td>+0.25</td>
<td>+0.10</td>
<td>+1.54</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>√</td>
<td>+0.21</td>
<td>+0.23</td>
<td>+0.08</td>
<td>+0.85</td>
<td></td>
</tr>
</tbody>
</table>

Table 3, three baseline models which utilize different PLMs to train the Transformer encoder of our proposed framework, are represented as “XLNET”, “BERT”, and “RoBERTA”, respectively. There are three observations drawn from the results. First, The framework which integrates with our proposed LbGCN outperforms the baseline models for all 5 datasets in terms of F-scores and for the majority of datasets in terms of Rs. Second, the proposed LbGCN makes small improvements in some datasets, whereas considerable improvements are shown in the other datasets. The extent of improvement of LbGCN does not depend on PLMs which the encoder utilizes. For instance, when training the Transformer encoder fine-tuning the RoBERTa, LbGCN improves the F-score on the PKU dataset from 97.00 to 97.21 and Rs from 89.80 to 90.03. With XLNET or BERT as the baseline PLM, the improvement of LbGCN on F-scores and Rs are still decent. Lastly, the methods that fine-tune the RoBERTa can achieve better performance on most benchmarks, and our proposed LbGCN utilizes the GCN to get further promotion on the baseline model which already achieves competitive performance of CWS.

Besides, we compare the proposed framework with existing methods. The comparison is also presented in Table 3, where the proposed framework LbGCN based on the BERT or RoBERTa outperforms all existing models in terms of the F-scores on all benchmarks.

Results on Cross-Domain CWS Domain variance is important to affect the performance of word segmenters. To demonstrate the efficiency of LbGCN, we also run frameworks with and without the LbGCN in the cross-domain scenario. Table 4 reports the results in F-score, which shows a similar trend as that in Table 3, where LbGCN outperforms baselines in all 5 domains. And the
framework ROBERTA-LBGCN achieves state-of-the-art performance in terms of average F-score. In particular, the XLNET has better performance on the “Computer” domain, and the BERT has better performance on the “Literature” domain. In general, our proposed LBGCN mechanism can effectively improve performance in the cross-domain scenario, and all LBGCNs fine-tuning different PLMs achieve competitive performance, compared with existing methods.

4.3 Effect of Using Different Lexicons

LBGCN utilizes a general way of integrating lexicon for CWS. To analyze the effect of methods using different lexicons, we adopt four different lexicons into the ROBERTA-LBGCN and compare them with the baseline model, as shown in Figure 2. Four lexicons consist of two simplified Chinese dictionaries\(^3\) and two traditional Chinese dictionaries\(^4\). Particularly, two simplified Chinese dictionaries consist of a basic version “LBGCNs” (red) and a modified version “LBGCNsdl” (yellow), respectively. Similarly, two traditional Chinese dictionaries are also a basic version “LBGCNt” (purple), and a modified version “LBGCNtd” (green).

As shown in Figure 2, the performance of using the four lexicons are all better than those of the baseline models on both the “PKU” and “AS” dataset, indicating the efficiency of our proposed lexicon-based framework. The framework using the basic simplified Chinese dictionary (red) achieves the biggest improvement on the “PKU” and the one using the basic traditional Chinese dictionary (purple) achieves the biggest improvement on the “AS”.

4.4 Ablation Study

LBGCN integrates two additional components for CWS, including the bi-gram features and the lexicon-based GCN. To analyze the effect of LBGCN with respect to different components, we do an ablation experiment based on the ROBERTA PLM which performs better for both “PKU” and “MSR” benchmarks and the results are shown in Table 5. Table 5 shows that the GCN (ID:3,4) effectively improves the performance of the baseline model on “PKU” and “MSR”, and it also alleviates the issue of OOV words, indicating the effectiveness of our proposed framework. While the GCN that integrates with the bi-gram component (ID:4) achieves progress on the “PKU” from +0.15 to +0.21, it hurts the Roov. A single bi-gram component (ID:2) hardly affects the F-score but it can improve the recall of OOV words. In terms of the results in Table 5, the bi-gram and GCN boost the performance considerably.

4.5 Case Study

To investigate how the proposed framework learns from the lexicon-based GCN, we choose an example input sentence “在青石板路上的清脆回声” (The clear echo on the flagstone road) in the literature domain scenario as a case study. In this sentence, the n-gram “青石板” (flagstone road) is the road that is made of a special kind of stone and always occurs in the Chinese literature. However, the split “板路” is short for the plate circuit. The baseline model may confuse this case because of the character diversity in Chinese. Intuitively, the “青石板” is in the pre-defined lexicon and an
undirected graph is constructed with the information of the word boundary. Then the lexicon-based GCN capture this information and integrates the graph embeddings with the original hidden states. The integrated embeddings with the knowledge of lexicon information are transferred into correct tags by the decoder. In Figure 3, we visualize the resulted weights that learn from the basic Transformer encoder (a), as well as from the final tagger (b).

In addition, in another case, “地瓜/粥” (sweet potato congee) represents a Chinese food and “粥” (congee) should be regarded as a single suffix word on “PKU” segmentation criterion. The baseline model cannot segment it correctly, because it keeps the superabundant pre-trained knowledge of PLMs. In the LBGCN, “地瓜粥” does not exist in the lexicon but “地瓜” is a lexicon word. The LBGCN constructs this relationship in the graph to distinguish important n-grams and improves performance accordingly for CWS.

5 Conclusion

To make up for the insufficiency of previous methods that fine-tune PLMs, in this paper, we propose a lexicon-based graph convolutional network to better transfer pre-trained knowledge from PLMs into the CWS. Our proposed framework LBGCN provides baseline models with the information of word boundary and contextual information, in addition to preserving the merits of baseline models in applying PLMs. In summary, the advantages of LBGCN are threefold. First, the novel framework does not rely on a particular PLM, and it can get further promotion on all baseline methods based on three mainstream PLMs for CWS. Second, the results on extensive experiments show that LBGCN achieves competitive performance on the CWS benchmarks, compared with previous methods. Third, further experiments and analyses demonstrate the effectiveness of LBGCN in the cross-domain scenario as well as when using different lexicons and components. Overall, this paper presents an elegant way to use a graph neural network for CWS and enhance fine-tuning CWS methods. For future work, we plan to investigate other sequence labeling tasks using the same methodology.

Acknowledgments

We sincerely thank the reviewers for their insightful comments and suggestions to improve the quality of the paper. The authors gratefully acknowledge the financial support provided by the National Key Research and Development Program of China (2020AAA0108004) and the National Natural Science Foundation of China under (No.U1936109).

References


