Joint Chinese Word Segmentation and Part-of-speech Tagging via Two-way Attentions of Auto-analyzed Knowledge

Yuanhe Tian[♡*], **Yan Song**[♠†], **Xiang Ao**[♣], **Fei Xia**[♡],
Xiaojun Quan[△], **Tong Zhang**[◇], **Yonggang Wang**[♠]
[♡]University of Washington, [♠]Sinovation Ventures
[♣]Chinese Academy of Sciences, [△]Sun Yat-sen University
[◇]The Hong Kong University of Science and Technology
[♡]{yhtian, fxia}@uw.edu [♠]clksong@gmail.com
[♣]aoxiang@ict.ac.cn [△]quanxj3@mail.sysu.edu.cn
[◇]tongzhang@ust.hk [♠]wangyonggang@chuangxin.com

Abstract

Chinese word segmentation (CWS) and part-of-speech (POS) tagging are important fundamental tasks for Chinese language processing, where joint learning of them is an effective one-step solution for both tasks. Previous studies for joint CWS and POS tagging mainly follow the character-based tagging paradigm with introducing contextual information such as n-gram features or sentential representations from recurrent neural models. However, for many cases, the joint tagging needs not only modeling from context features but also knowledge attached to them (e.g., syntactic relations among words); limited efforts have been made by existing research to meet such needs. In this paper, we propose a neural model named TwASP for joint CWS and POS tagging following the character-based sequence labeling paradigm, where a two-way attention mechanism is used to incorporate both context feature and their corresponding syntactic knowledge for each input character. Particularly, we use existing language processing toolkits to obtain the auto-analyzed syntactic knowledge for the context, and the proposed attention module can learn and benefit from them although their quality may not be perfect. Our experiments illustrate the effectiveness of the two-way attentions for joint CWS and POS tagging, where state-of-the-art performance is achieved on five benchmark datasets.[1]

1 Introduction

Chinese word segmentation (CWS) and part-of-speech (POS) tagging are two fundamental and crucial tasks in natural language processing (NLP) for Chinese. The former one aims to find word boundaries in a sentence and the latter, on the top of segmentation results, assigns a POS tag to each word to indicate its syntactical property in the sentence. To effectively perform CWS and POS tagging, combining them into a joint task is proved to have better performance than separately conducting the two tasks in a sequence (Ng and Low, 2004). Therefore, many studies were proposed in the past decade for joint CWS and POS tagging (Jiang et al., 2008, 2009; Sun, 2011; Zeng et al., 2013; Zheng et al., 2013; Kurita et al., 2017; Shao et al., 2017; Zhang et al., 2018). These studies, regardless of whether they used conventional approaches (Jiang et al., 2008, 2009; Sun, 2011; Zeng et al., 2013) or deep learning based approaches (Zheng et al., 2013; Kurita et al., 2017; Shao et al., 2017; Zhang et al., 2018), focused on incorporating contextual information into their joint tagger.

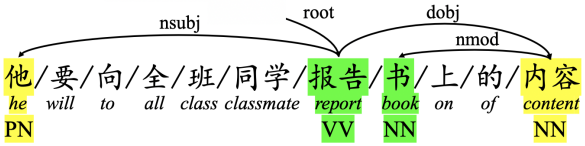

Figure 1: An example sentence with CWS and POS tagging results, where the ambiguous part (in green color) has dependencies from distant words (in yellow color).

In addition, it is well known that syntactic structure is also able to capture and provide the information of long-distance dependencies among words. For example, Figure 1 shows an example of local ambiguity, where the green highlighted part has two possible interpretations – "报告_VV/书_NN" (*report a book*) and "报告书_NN" (*the report*). The ambiguity can be resolved with syntactic analysis; for instance, the dependency structure, if available, would prefer the first interpretation. While the subject and the object of the sentence (highlighted in yellow) are far away from the ambiguous part in

[*]Partially done as an intern at Sinovation Ventures.
[†]Corresponding author.
[1]TwASP (code and the best performing models) is released at https://github.com/SVAIGBA/TwASP.

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 8286–8296
July 5 - 10, 2020.

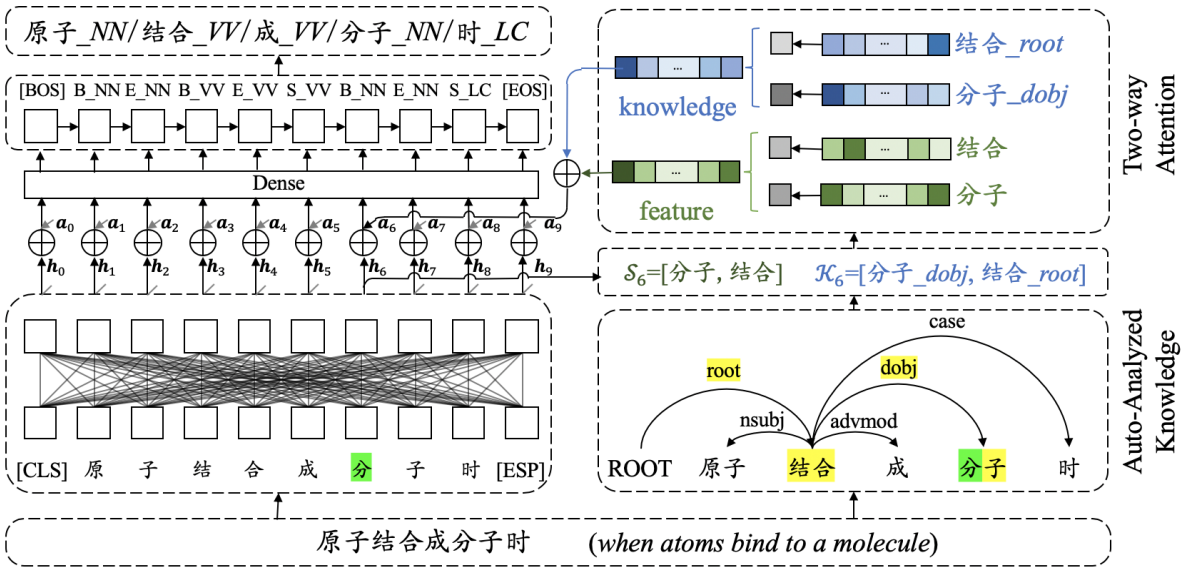

Figure 2: The architecture of TwASP for the joint CWS and POS tagging with the two-way attention mechanism, which is presented with example context features and their dependency knowledge (highlighted in yellow) from auto-analyzed results for a character (i.e., "分" (*split*) highlighted in green) in the given sentence.

the surface word order, they are much closer in the dependency structure (the subject depends on "报告_*VV*" and "书_*NN*" depends on the the object). This example shows that syntactic structure provides useful cues for CWS and POS tagging.

Syntactic knowledge can be obtained from manually constructed resources such as treebanks and grammars, but such resources require considerate efforts to create and might not be available for a particular language or a particular domain. A more practical alternative is to use syntactic structures automatically generated by off-the-shelf toolkits. Some previous studies (Huang et al., 2007; Jiang et al., 2009; Wang et al., 2011; Zhang et al., 2018) verified the idea for this task by learning from auto-processed corpora. However, their studies treat auto-processed corpora as gold reference and thus are unable to distinguishingly use it according to its quality (the resulted knowledge is not accurate in most cases). Therefore, the way to effectively leverage such auto-generated knowledge for the joint CWS and POS tagging task is not fully explored.

In this paper, we propose a neural model named TwASP with a two-way attention mechanism to improve joint CWS and POS tagging by learning from auto-analyzed syntactic knowledge, which are generated by existing NLP toolkits and provide necessary (although not perfect) information for the task. In detail, for each input character, the proposed attention module extracts the context features associated with the character and their corresponding knowledge instances according to the auto-analyzed results, then computes the attentions separately for features and knowledge in each attention way, and finally concatenates the attentions from two ways to guide the tagging process. In doing so, our model can distinguish the important auto-analyzed knowledge based on their contributions to the task and thus avoid being influenced by some inferior knowledge instances. Compared to another prevailing model, i.e., key-value memory networks (Miller et al., 2016), which can learn from pair-wisely organized information, the two-way attentions not only are able to do so, but also fully leverage features and their knowledge rather than using one to weight the other.[2] We experiment with three types of knowledge, namely, POS labels, syntactic constituents, and dependency relations, in our experiments. The experimental results on five benchmark datasets illustrate the effectiveness of our model, where state-of-the-art performance for the joint task is achieved on all datasets. We also perform several analyses, which confirm the validity of using two-way attentions and demonstrate that our model can be further improved by synchronously using multiple types of knowledge.

2 The Model

The architecture of TwASP is illustrated in Figure 2. The left part shows the backbone of the model for the joint CWS and POS tagging following

[2] We explain it in later part of the paper that, the output of key-value memory networks mainly rely on the value embeddings, where keys are used to weight such embeddings.

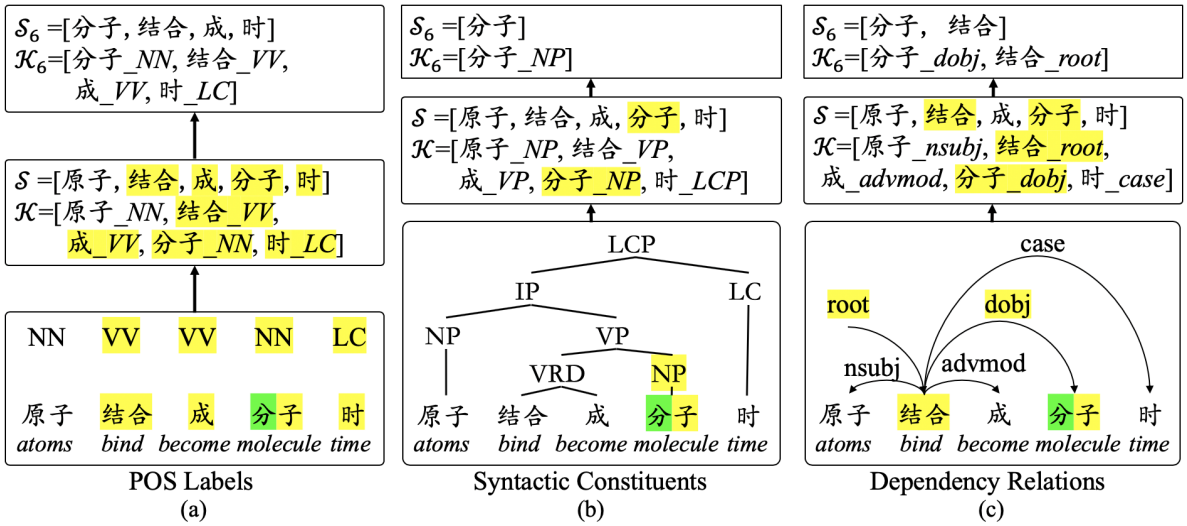

Figure 3: Examples of context features and their corresponding knowledge from (a) POS labels, (b) syntactic constituents and (c) dependency relations. Features and knowledge for the character "分" are highlighted in yellow.

the character-based sequence labeling paradigm, where the input is a character sequence $\mathcal{X} = x_1 x_2 \cdots x_i \cdots x_l$ and the output is a sequence of joint labels $\mathcal{Y} = y_1 y_2 \cdots y_i \cdots y_l$. To enhance the backbone paradigm, the proposed two-way attention module (as shown in the right part of Figure 2) takes the syntactic knowledge produced from the input sentence, analyzes it and then feeds it to the tagging process. In this section, we firstly introduce the auto-analyzed knowledge, then explain how the two-way attentions consume such knowledge, and finally describe how the joint CWS and POS tagging works with the resulted attentions.

2.1 Auto-analyzed Knowledge

Auto-analyzed knowledge is demonstrated to be an effective type of resources to help NLP systems understand the texts (Song et al., 2017; Seyler et al., 2018; Huang and Carley, 2019). One challenge for leveraging external knowledge for the joint task is that gold-standard annotations are extremely rare for text in most domains, especially the syntactic annotations. An alternative solution is to use off-the-shelf NLP systems to produce such knowledge, which is proved to be useful in previous studies (Huang et al., 2007; Jiang et al., 2009; Wang et al., 2011; Zhang et al., 2018). Rather than processing an entire corpus and then extracting features or training embeddings from the resulted corpus as in previous studies, our model does not treat knowledge as gold references: it generates auto-analyzed knowledge for each sentence and learns the weights of the corresponding features. Formally, for a character sequence $\mathcal{X}$, let $\mathcal{S}$ and $\mathcal{K}$ denote the lists of context features and knowledge for $\mathcal{X}$, respectively. For each character x_i in $\mathcal{X}$, let $\mathcal{S}_i = [s_{i,1}, s_{i,2}, \cdots s_{i,j}, \cdots s_{i,m_i}]$ and $\mathcal{K}_i = [k_{i,1}, k_{i,2}, \cdots k_{i,j}, \cdots k_{i,m_i}]$ be the sublists of $\mathcal{S}$ and $\mathcal{K}$ for x_i. Here, $s_{i,j}$ and $k_{i,j}$ denote a context feature and a knowledge instance, respectively.

In this paper, we use three types of syntactic knowledge for the joint task, namely POS labels, syntactic constituents, and dependency relations, where POS labels indicate the syntactic information of individual words, syntactic constituents provide the structural grouping information for a text span, and dependencies offer dependency relations between words. Figure 3 shows an example sentence and the corresponding $\mathcal{S}$ and $\mathcal{K}$. For character "分" (highlighted in green), its $\mathcal{S}_i$ and $\mathcal{K}_i$ are highlighted in yellow. In order to distinguish same knowledge appearing with different context features, we use a feature-knowledge combination tag to represent each knowledge instance (e.g., "分子_*NN*", "分子_*NP*", and "分子_*dobj*" in Figure 3). We explain each type of knowledge below.

POS Labels Figure 3 (a) shows that, for each x_i (e.g., x_6 ="分"), we use a 2-word window for both sides to extract context features from $\mathcal{S}$ to form $\mathcal{S}_i$ (i.e., $\mathcal{S}_6$ = ["分子", "结合", "成", "时"]), and then get their corresponding knowledge instances of POS labels from $\mathcal{K}$ to form $\mathcal{K}_i$ (i.e., $\mathcal{K}_6$ = ["分子_*NN*", "结合_*VV*", "成_*VV*", "时_*LC*"]).

Syntactic Constituents As shown in Figure 3 (b), the rule for extracting syntactic constituency knowledge is as follows. We start with the word containing the given character x_i, go up the con-

stituency tree to the first ancestor whose label is in a pre-defined syntactic label list,[3] then use all the words under this node to select context features from $\mathcal{S}$, and finally combine the words with the syntactic label of the node to select knowledge instances from $\mathcal{K}$. For example, for x_6="分", the lowest syntactic node governing "分子" is *NP* (highlighted in yellow); thus $\mathcal{S}_6$ = ["分子"] and $\mathcal{K}_6$ = ["分子_*NP*"]. Another example is x_5="成", the lowest acceptable node on its syntactic path is *VP*; therefore, $\mathcal{S}_5$ = ["结合", "成", "分子"] and $\mathcal{K}_5$ = ["结合_*VP*", "成_*VP*", "分子_*VP*"].

Dependency Relations Given a character x_i, let w_i be the word that contains x_i. The context features $\mathcal{S}_i$ include w_i, w_i's governor, and w_i's dependents in the dependency structure; those words combined with their inbound dependency relation labels form $\mathcal{K}_i$. For example, for x_6="分", w_6 = "分子", which depends on "结合" with a dependency label *dobj*. Therefore, $\mathcal{S}_6$ = ["分子", "结合"], and $\mathcal{K}_6$ = ["分子_*obj*", "结合_*root*"].

2.2 Two-Way Attentions

Attention has been shown to be an effective method for incorporating knowledge into NLP systems (Kumar et al., 2018; Margatina et al., 2019) but it cannot be used directly for feature and knowledge in pair-wise forms. Previous studies on the joint task normally directly concatenate the embeddings from context features and knowledge instances into the embeddings of characters (Zhang et al., 2018), which could be problematic for incorporating auto-analyzed, error-prone syntactic knowledge obtained from off-the-shelf toolkits.

For both features and their knowledge instances for $\mathcal{X}$, we use a two-way attention design to have separate attention for $\mathcal{S}$ and $\mathcal{K}$. Particularly, the two ways, namely, the feature way and the knowledge way, are identical in architecture, where each way has a feed-forward attention module (Raffel and Ellis, 2015). For each x_i, its $\mathcal{S}_i$ and $\mathcal{K}_i$ are firstly fed into the feature attention way and the knowledge attention way, respectively, then computed within each way, and their final attention vectors are combined to feedback to the backbone model.

Take the feature way as an example, the attention weight for each context feature $s_{i,j}$ is computed by

$$a^s_{i,j} = \frac{exp(\mathbf{h}_i^\top \cdot \mathbf{e}^s_{i,j})}{\sum_{j=1}^{m_i} exp(\mathbf{h}_i^\top \cdot \mathbf{e}^s_{i,j})} \quad (1)$$

where $\mathbf{h}_i$ is the vector from a text encoder for x_i and $\mathbf{e}^s_{i,j}$ the embedding of $s_{i,j}$. Then we have the weighted embedding $\mathbf{a}^s_i$ for all $s_{i,j}$ in $\mathcal{S}_i$ via

$$\mathbf{a}^s_i = \sum_{j=1}^{m_i} a^s_{i,j}\mathbf{e}^s_{i,j} \quad (2)$$

where $\sum$ denotes a element-wise sum operation.

For the knowledge way, the same process is applied to get $\mathbf{a}^k_i$ by distinguishing and weighting each knowledge instance $k_{i,j}$. Finally, the output of the two attention ways are obtained through an concatenation of the two vectors: $\mathbf{a}_i = \mathbf{a}^s_i \oplus \mathbf{a}^k_i$.

2.3 Joint Tagging with Two-way Attentions

To functionalize the joint tagging, the two-way attentions interact with the backbone model through the encoded vector $\mathbf{h}_i$ and its output $\mathbf{a}_i$ for each x_i.

For $\mathbf{h}_i$, one can apply many prevailing encoders, e.g., Bi-LSTM or BERT (Devlin et al., 2019), to get the vector list $[\mathbf{h}_1\mathbf{h}_2 \cdots \mathbf{h}_i \cdots \mathbf{h}_l]$ for $\mathcal{X}$.

Once $\mathbf{a}_i$ is obtained, we concatenate it with $\mathbf{h}_i$ and send it through a fully connected layer to align the dimension of the output for final prediction:

$$\mathbf{o}_i = \mathbf{W} \cdot (\mathbf{h}_i \oplus \mathbf{a}_i) + \mathbf{b} \quad (3)$$

where $\mathbf{W}$ and $\mathbf{b}$ are trainable parameters. Afterwards, conditional random fields (CRF) is used to estimate the probability for y_i over all possible joint CWS and POS tags under x_i and y_{i-1} by

$$p(y_i|x_i) = \frac{exp(\mathbf{W}_c \cdot \mathbf{o}_i + \mathbf{b}_c)}{\sum_{y_{i-1}y_i} exp(\mathbf{W}_c \cdot \mathbf{o}_i + \mathbf{b}_c)} \quad (4)$$

Here, $\mathbf{W}_c$ and $\mathbf{b}_c$ are the weight matrix and the bias vector, respectively, and they are estimated using the (y_{i-1}, y_i) tag pairs in the gold standard.

3 Experiments

3.1 Datasets

We employ five benchmark datasets in our experiments, where four of them, namely, CTB5, CTB6, CTB7, and CTB9, are from the Penn Chinese TreeBank[4] (Xue et al., 2005) and the fifth one is

[3] Following Chen et al. (2006), the list has 12 syntactic labels, namely, ADJP, ADVP, CLP, DNP, DP, DVP, LCP, LST, NP, PP, QP, and VP.

[4] We obtain the Penn Chinese TreeBank data from the official release of Linguistic Data Consortium. The catalog numbers for CTB5, CTB6, CTB7, and CTB9 are LDC2005T01, LDC2007T36, LDC2010T07, and LDC2016T13, respectively.

Datasets		Char	Word	Sent	OOV %
CTB5	Train	805K	494K	18K	-
	Dev	12K	7K	350	8.1
	Test	14K	8K	348	3.5
CTB6	Train	1,056K	641K	23K	-
	Dev	100K	60K	2K	5.4
	Test	134K	82K	3K	5.6
CTB7	Train	1,160K	718K	31K	-
	Dev	387K	237K	10K	5.5
	Test	399K	245K	10K	5.2
CTB9 (general)	Train	2,643K	1,696K	106K	-
	Dev	210K	136K	10K	2.9
	Test	379K	242K	16K	3.1
UD	Train	156K	99K	4K	-
	Dev	20K	13K	500	12.1
	Test	19K	12K	500	12.4
CTB9 (genres)	BC	275K	184K	12K	2.8
	BN	483K	287K	10K	5.1
	CS	228K	160K	17K	5.5
	DF	644K	421K	20K	3.7
	MZ	403K	258K	8K	7.5
	NW	427K	251K	10K	5.1
	SC	430K	304K	44K	4.0
	WB	342K	210K	10K	5.3

Table 1: The statistics of all experimental datasets in terms of character, word and sentence numbers. For normal splits, OOV % is computed according to the training set; for each genre in CTB9, OOV % is computed with respect to the union of other seven genres.

the Chinese part of Universal Dependencies (UD)[5] (Nivre et al., 2016). The CTB datasets are in simplified Chinese characters while the UD dataset is in traditional Chinese. Following Shao et al. (2017), we convert the UD dataset into simplified Chinese[6] before conducting experiments on it.

CTB uses 33 POS tags, and we split CTB5-CTB9 following previous studies (Wang et al., 2011; Jiang et al., 2008; Shao et al., 2017). In addition, because the data in CTB9 come from eight genres – broadcast conversation (BC), broadcast news (BN), conversational speech(CS), discussion forums (DF), magazine articles (MZ), newswire (NW), SMS/chat messages (SC), and weblog (WB) – we also use CTB9 in a cross-domain study (see Section 3.4). UD uses two POS tagsets, namely the universal tagset (15 tags) and language-specific tagset (42 tags for Chinese). We refer to the corpus with the two tagsets as UD1 and UD2, respectively, and use the official splits of train/dev/test in our experiments. The statistics for the aforementioned datasets are in Table 1.

[5]We use its version 2.4 downloaded from https://universaldependencies.org/.

[6]The conversation scripts are from https://github.com/skydark/nstools/tree/master/zhtools

			CTB5	CTB6	CTB7	CTB9	UD
$\mathcal{S}$			20K	23K	24K	41K	7K
$\mathcal{K}$	SCT	POS	22K	25K	27K	46K	7K
		Syn.	70K	82K	87K	141K	31K
		Dep.	32K	39K	42K	77K	8K
	BNP	POS	22K	26K	28K	48K	8K
		Syn.	69K	81K	85K	136K	29K

Table 2: Numbers of context features ($\mathcal{S}$) and their corresponding knowledge instances ($\mathcal{K}$) for five benchmark datasets, based on the output of SCT and BNP. Note that the $\mathcal{K}$ for the UD dataset follows the CTB criteria, because SCT and BNP were trained on CTB.

3.2 Implementation

To obtain the aforementioned three types of knowledge, we use two off-the-shelf toolkits, Stanford CoreNLP Toolkit (SCT)[7] (Manning et al., 2014) and Berkeley Neural Parser (BNP)[8] (Kitaev and Klein, 2018): the former tokenizes and parses a Chinese sentence, producing POS tags, phrase structure and dependency structure of the sentence; the latter does POS tagging and syntactic parsing on a pre-tokenized sentence. Both toolkits were trained on CTB data and thus produced CTB POS tags. To extract knowledge, we firstly use SCT to automatically segment sentences and then run both SCT and BNP for POS tagging and parsing. Table 2 shows the size of $\mathcal{S}$ and $\mathcal{K}$ for all the datasets.

We test the model with three encoders: two of them, namely Bi-LSTM and BERT[9] (Devlin et al., 2019), are widely used; the third encoder is ZEN[10] (Diao et al., 2019), which is a recently released Chinese encoder pre-trained with n-gram information and outperforms BERT in many downstream tasks. For the Bi-LSTM encoder, we set its hidden state size to 200 and use the character embeddings released by Shao et al. (2017) to initialize its input representations. For BERT and ZEN, we follow their default settings, e.g., 12 layers of self-attentions with the dimension of 768.

For the two-way attention module, we randomly initialize the embeddings for all context features and their corresponding knowledge instances, where one can also use pre-trained embeddings (Song et al., 2018; Grave et al., 2018; Zhang et al., 2019; Yamada et al., 2020) for them. For all the

[7]We use its version 3.9.2 downloaded from https://stanfordnlp.github.io/CoreNLP/.

[8]We download the model from https://github.com/nikitakit/self-attentive-parser.

[9]We use the Chinese base model from https://s3.amazonaws.com/models.huggingface.co/.

[10]https://github.com/sinovation/ZEN

	CTB5		CTB6		CTB7		CTB9		UD1		UD2	
	Seg	Joint	Seg	Joint	Seg	Joint	Seg	Joint	Seg	Joint	Seg	Joint
SCT	98.02	95.49	96.62	90.85	96.53	92.73	93.63	88.23	80.50*	0.00*	80.50*	36.11*
BNP	-	95.50	-	94.43	-	92.95	-	88.09	-	0.00*	-	37.16*
Bi-LSTM	97.69	93.73	95.46	90.63	95.46	89.98	96.45	91.80	94.96	88.72	95.01	88.75
+ POS (SCT)	**98.07**	94.68	96.23	91.04	96.32	91.60	96.75	92.36	94.86	88.90	**95.08**	88.99
+ Syn. (SCT)	98.03	**95.66**	96.06	90.97	95.90	91.90	96.57	92.40	94.88	88.87	94.71	88.90
+ Dep. (SCT)	97.84	94.25	95.85	90.70	95.87	91.08	96.63	92.26	94.88	88.93	94.91	89.05
+ POS (BNP)	98.06	95.34	**96.46**	**93.31**	**96.58**	**92.87**	**96.73**	**93.38**	**95.02**	**89.27**	94.89	89.17
+ Syn. (BNP)	98.01	94.82	96.08	92.33	96.06	91.04	96.65	92.97	94.48	88.84	94.86	**89.20**
BERT	98.28	96.03	97.36	94.65	96.78	93.38	97.33	94.40	97.74	94.82	97.70	94.76
+ POS (SCT)	**98.77**	**96.77**	97.43	94.82	97.31	94.12	**97.75**	**94.87**	**98.32**	**95.60**	**98.33**	**95.46**
+ Syn. (SCT)	98.75	96.66	97.37	94.73	97.07	93.84	97.67	94.83	98.11	95.43	98.10	95.42
+ Dep. (SCT)	98.65	96.69	97.35	94.87	97.10	93.89	97.67	94.82	98.10	95.41	98.11	95.36
+ POS (BNP)	98.63	96.60	97.34	94.95	97.25	94.21	97.65	94.82	98.16	95.51	98.22	95.23
+ Syn. (BNP)	98.75	96.72	**97.39**	**94.99**	**97.32**	**94.28**	97.69	94.85	98.25	95.42	98.17	95.18
ZEN	98.61	96.60	97.35	94.70	97.09	93.80	97.64	94.64	98.14	95.15	98.02	95.05
+ POS (SCT)	98.81	**96.92**	97.45	94.87	97.27	94.20	**97.77**	**94.88**	**98.33**	**95.69**	98.18	95.49
+ Syn. (SCT)	**98.85**	96.86	97.42	94.72	**97.31**	**94.32**	97.73	94.85	98.17	95.48	**98.35**	**95.50**
+ Dep. (SCT)	98.82	96.85	97.38	94.75	97.25	94.22	97.70	94.85	98.27	95.68	98.28	95.32
+ POS (BNP)	98.72	96.83	**97.47**	**95.02**	97.24	94.18	97.69	94.82	98.26	95.52	98.22	95.28
+ Syn. (BNP)	98.83	96.83	97.44	94.95	97.25	94.18	97.67	94.86	98.22	95.49	98.20	95.45

Table 3: Experimental results (the F-scores for segmentation and joint tagging) of TWASP using different encoders with and without auto-analyzed knowledge on the five benchmark datasets. "Syn." and "Dep." refer to syntactic constituents and dependency relations, respectively. The results of SCT and BNP are also reported as references, where * marks that the segmentation and POS tagging criteria from the toolkits and the UD dataset are different.

models, we set the maximum character length of the input sequence to 300 and use negative log-likelihood loss function. Other hyper-parameters of the models are tuned on the dev set and the tuned models are evaluated on the test set for each dataset (each genre for CTB9). F-scores for word segmentation and the joint CWS-POS tags are used as main evaluation metrics[11] in all experiments.

3.3 Overall Performance

In our main experiment, we run our TWASP on the five benchmark datasets using the three encoders, i.e., Bi-LSTM, BERT, and ZEN. The results on the F-scores of word segmentation and joint CWS and POS tagging are in Table 3, which also includes the performance of the baselines without attention and the two toolkits (i.e., SCT and BNP). The results of SCT and BNP on the UD dataset are bad because they were trained on CTB, which used different segmentation and POS tagging criteria.

There are several observations. First, for all encoders, the two-way attentions provide consistent enhancement to the baselines with different types of knowledge. Particularly, although the baseline model is well-performed when BERT (or ZEN) serves as the encoder, the attention module is still able to further improve its performance with the knowledge produced by the toolkits even though the toolkits have worse-than-baseline results for the joint task. Second, among different types of knowledge, POS labels are the most effective ones that help the joint task. For instance, among BERT-based models, the one enhanced by POS knowledge from SCT achieves the best performance on most datasets, which is not surprising because such knowledge matches the outcome of the task. In addition, for BERT-based models enhanced by knowledge from BNP (i.e., BERT + POS (BNP) and BERT + Syn. (BNP)), syntactic constituents provide more improvement than POS labels on all CTB datasets. This observation could be explained by that BNP is originally designed for constituency parsing with CTB criteria; the syntactic constituents are complicated while effective when they are accurate. Third, while SCT and BNP were trained on CTB, whose tagset is very different from the two tagsets for UD, TWASP still outperforms the baselines on UD with the knowledge provided by SCT and BNP, indicating that syntactic knowledge is useful even when it uses different word segmentation and POS tagging criteria.

Table 4 shows the results of our best models (i.e. BERT and ZEN with POS (SCT)) and previous studies on the same datasets. Our approach

[11] We use the evaluation script from https://github.com/chakki-works/seqeval.

	CTB5		CTB6		CTB7		CTB9		UD1		UD2	
	Seg	Joint	Seg	Joint	Seg	Joint	Seg	Joint	Seg	Joint	Seg	Joint
Jiang et al. (2008)	97.85	93.41	-	-	-	-	-	-	-	-	-	-
Kruengkrai et al. (2009)	97.87	93.67	-	-	-	-	-	-	-	-	-	-
Sun (2011)	98.17	94.02	-	-	-	-	-	-	-	-	-	-
Wang et al. (2011)	98.11	94.18	95.79	91.12	95.65	90.46	-	-	-	-	-	-
Qian and Liu (2012)	97.85	93.53	-	-	-	-	-	-	-	-	-	-
Shen et al. (2014)	98.03	93.80	-	-	-	-	-	-	-	-	-	-
Kurita et al. (2017)	98.41	94.84	-	-	96.23	91.25	-	-	-	-	-	-
Shao et al. (2017)	98.02	94.38	-	-	-	-	96.67	92.34	95.16	89.75	95.09	89.42
Zhang et al. (2018)	98.50	94.95	96.36	92.51	96.25	91.87	-	-	-	-	-	-
BERT + POS (SCT)	98.77	96.77	97.43	94.82	**97.31**	94.12	97.75	94.87	98.32	95.60	**98.33**	95.46
ZEN + POS (SCT)	**98.81**	**96.92**	**97.45**	**94.87**	97.27	**94.20**	**97.77**	**94.88**	**98.33**	**95.69**	98.18	**95.49**

Table 4: Comparison (in F-scores of word segmentation and joint tagging) of TWASP (with BERT or ZEN encoder) with previous studies. Cells with "-" refer to the results are not reported or they are not applicable.

outperforms previous studies on the joint task and achieves new state-of-the-art performance on all datasets. While some of the previous studies use auto-analyzed knowledge (Wang et al., 2011; Zhang et al., 2018), they regard such knowledge as gold reference and consequently could suffer from errors in the auto-analyzed results. In contrast, our proposed model is able to selectively model the input information and to discriminate useful knowledge instances through the two-way attentions.

3.4 Cross-Domain Performance

Domain variance is an important factor affecting the performance of NLP systems (Guo et al., 2009; McClosky et al., 2010; Song and Xia, 2013). To further demonstrate the effectiveness of TWASP, we conduct cross-domain experiments on the eight genres of CTB9 using BERT and ZEN as the baseline and their enhanced version with POS knowledge from SCT. In doing so, we test on each genre with the models trained on the data from all other genres. The results on both segmentation and the joint task are reported in Table 5, where the SCT results are also included as a reference.

The comparison between the baselines and TWASP with POS knowledge clearly shows the consistency of performance improvement with two-way attentions, where for both BERT and ZEN, TWASP outperforms the baselines for all genres on the joint labels. In addition, similar to the observations from the previous experiment, both accurate and inaccurate POS knowledge are able to help the joint task. For example, although the SCT results on several genres (e.g., CS, DF, SC) are much worse than of the BERT baseline, the POS labels produced by SCT can still enhance TWASP on word segmentation and joint tagging with the proposed two-way attention module.

4 Analysis

4.1 The Effect of Two Attention Ways

In the first analysis, we compare our two-way attention with normal attention. For normal attention, we experiment three ways of incorporating context features and knowledge: (1) using context features and knowledge together in the attention, where all features or knowledge instances are equally treated in it; (2) using context features only; and (3) using knowledge only. We run these experiments with BERT encoder and POS knowledge from SCT on CTB5 and report the results in Table 6. Overall, the two-way attentions outperform all three settings for normal attention, which clearly indicates the validity of using two attention ways for features and knowledge, i.e., compared to (1), as well as the advantage of learning from both of them, i.e., compared to (2) and (3). Interestingly, in the three settings, (3) outperforms (1), which could be explained by that, with normal attention, mixed feature and knowledge instances in it may make it difficult to weight them for the joint task.

There are other methods for using both context features and knowledge in a neural framework, such as key-value memory networks (kvMN) (Miller et al., 2016), which is demonstrated to improve CWS by Tian et al. (2020). Thus we compare our approach with kvMN, in which context features are mapped to keys and knowledge to values. We follow the standard protocol of the kvMN, e.g., addressing keys by $\mathcal{S}_i$ and reading values from $\mathcal{K}_i$ through the corresponding knowledge for each key, computing weights from all key embeddings, and outputting the weighted embeddings from all values. The result from the kvMN is reported at the last row of Table 6, where its performance is not as good as the two-way attentions, and even

Genre	SCT		BERT		BERT+POS		ZEN		ZEN+POS	
	Seg	Joint	Seg	Joint	Seg	Joint	Seg	Joint	Seg	Joint
BC	96.27	93.55	96.29	92.08	96.38	92.34	96.48	92.25	**96.63**	92.41
BN	96.98	93.98	96.93	93.73	97.20	94.02	97.05	93.91	**97.21**	**94.14**
CS	89.83	81.93	95.17	89.18	95.14	89.46	95.10	89.24	**95.87**	**89.67**
DF	91.34	84.28	**96.79**	92.02	96.44	92.44	96.33	92.11	96.55	**92.51**
MZ	95.69	91.99	95.62	91.97	**95.83**	92.17	95.69	92.00	95.78	**92.18**
NW	97.41	94.75	97.55	94.44	97.49	94.64	97.49	94.51	**97.57**	94.70
SC	84.87	76.55	95.97	91.13	96.27	91.77	96.09	91.47	**96.38**	**91.85**
WB	95.99	92.86	95.09	89.59	95.11	89.85	95.10	89.74	95.35	90.10

Table 5: Experimental results (the F-scores for word segmentation and joint tagging) from baselines and TwASP with different encoders on eight genres of CTB9. The incorporated knowledge is the POS labels from SCT.

Ways		Seg		Joint
Feature	Knowledge	F	R_{OOV}	F
√	√	98.55	87.28	96.62
√	×	98.67	87.38	96.50
×	√	98.71	88.17	96.69
Two-way Attentions		**98.77**	88.13	**96.77**
Key-value Memory		98.62	**88.51**	96.58

Table 6: Performance comparison among different ways of knowledge integration, including normal attention (with respect to what knowledge type is used), the two-way attention, and key-value memory networks.

worse than using normal attention with setting (3). The reason could be straightforward: the output of kvMN is built upon value (knowledge) embeddings and therefore information from key (context feature) embeddings does not directly contribute to it other than providing weights for the value. As a result, kvMN acts in a similar yet inferior[12] way of setting (3) where only knowledge is used.

4.2 Knowledge Ensemble

Since every type of knowledge works well in our model, it is expected to investigate how the model performs when multiple types of knowledge are used together. To this end, we run experiments on CTB5 to test on our BERT-based TwASP with knowledge ensemble, where two ensemble strategies, i.e., averaging and concatenation, are applied with respect to how $\mathbf{a}_i$ for each knowledge type is combined with others. The results are reported in Table 7. In this table, the first seven rows (ID: 1-7) indicate that different types of knowledge are

[12]The "inferior" is explained by that, in kvMN, the value weights are inaccurate because they are computed with respect to the contribution of keys rather than knowledge instances.

ID	SCT			BNP		Joint F	
	POS	Syn.	Dep.	POS	Syn.	$\sum$	$\bigoplus$
1	√	√				96.79	96.80
2	√		√			96.78	96.81
3		√	√			96.79	96.80
4	√	√	√			96.82	96.87
5				√	√	96.76	96.81
6	√			√		96.81	96.83
7		√	√		√	96.82	96.84
8	√	√	√	√	√	96.87	**96.90**

Table 7: Comparison of different knowledge ensemble results, which are presented by the joint tagging F-scores from our BERT-based TwASP on CTB5. $\sum$ and $\bigoplus$ refer to averaging and concatenation of attentions from different knowledge types, respectively. As a reference, the best result on CTB5 for BERT-based model without knowledge ensemble is 96.77% achieved by BERT + POS (SCT) (see Table 3).

combined according to whether they come from the same toolkit (ID: 1-5) or belong to the same category (ID: 6 and 7); and the last row (ID: 8) is for the case that all types of knowledge are combined.

There are several observations. First, compared to only using one type of knowledge (refer to Table 3), knowledge ensemble improves model performance where more knowledge types contribute to better results. The best model is thus obtained when all knowledge (from each toolkit and from both toolkits) are used. Second, knowledge in the same type from different toolkits may complement to each other and thus enhance model performance accordingly, which is confirmed by the results from the models assembling POS (or Syn+Dep) information from both SCT and BNP. Third, for different ensemble strategies, concatenation tends to perform better than averaging, which is not surprising since concatenation actually turns the model into a multi-way structure for knowledge integration.

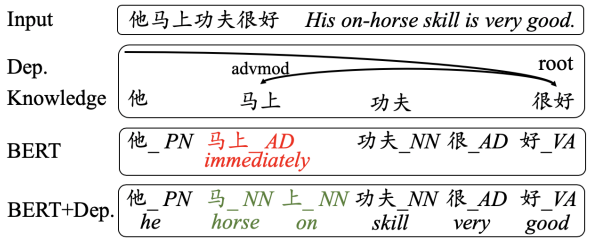

Figure 4: Comparison of joint tagging results between BERT and BERT+Dep (SCT) on an example sentence.

4.3 Case Study

When the toolkit provides accurate knowledge, it is not surprising that our two-way attention model would benefit from the auto-analyzed knowledge. Interestingly, even when the toolkit provides inaccurate output, our model might still be able to benefit from such output. Figure 4 shows such an example, where our system uses BERT+Dep using SCT and the baseline system is BERT without two-way attention. The sentence contains an ambiguity character bigram "马上", which has two possible interpretations, "马上_AD" (*immediately*) and "马_NN/上_LC" (*on the horse*). The second one is correct, yet the baseline tagger chooses the former because "马上" (*immediately*) is a very common adverb. Although SCT also chooses the wrong segmentation and thus has an incorrect dependency structure, our system is still able to produce correct segmentation and POS tags. One plausible explanation for this is that the inaccurate dependency structure includes an *advmod* link between "马上" (*immediately*) and "很好"(*very good*). Because such a dependency pair seldom appears in the corpus, the attention from such knowledge is weak and hence encourages our system to choose the correct word segmentation and POS tags.

5 Related Work

There are basically two approaches to CWS and POS tagging: to perform POS tagging right after word segmentation in a pipeline, or to conduct the two tasks simultaneously, known as joint CWS and POS tagging. In the past two decades, many studies have shown that joint tagging outperforms the pipeline approach (Ng and Low, 2004; Jiang et al., 2008, 2009; Wang et al., 2011; Sun, 2011; Zeng et al., 2013). In recent years, neural methods started to play a dominant role for this task (Zheng et al., 2013; Kurita et al., 2017; Shao et al., 2017; Zhang et al., 2018), where some of them tried to incorporate extra knowledge in their studies. For example, Kurita et al. (2017) exploited to model n-grams to improve the task; Shao et al. (2017) extended the idea by incorporating pre-trained n-gram embeddings, as well as radical embeddings, into character representations. Zhang et al. (2018) tried to leverage the knowledge from character embeddings, trained on an automatically tagged corpus by a baseline tagger. Compared to these previous studies, TwASP provides a simple, yet effective, neural model for joint tagging, without requiring a complicated mechanism of incorporating different features or pre-processing a corpus.

6 Conclusion

In this paper, we propose neural approach with a two-way attention mechanism to incorporate auto-analyzed knowledge for joint CWS and POS tagging, following a character-based sequence labeling paradigm. Our proposed attention module learns and weights context features and their corresponding knowledge instances in two separate ways, and use the combined attentions from the two ways to enhance the joint tagging. Experimental results on five benchmark datasets illustrate the validity and effectiveness of our model, where the two-way attentions can be integrated with different encoders and provide consistent improvements over baseline taggers. Our model achieves state-of-the-art performance on all the datasets. Overall, this work presents an elegant way to use auto-analyzed knowledge and enhance neural models with existing NLP tools. For future work, we plan to apply the same methodology to other NLP tasks.

Acknowledgement

Xiang Ao was partially supported by the National Natural Science Foundation of China under Grant No. 61976204, U1811461, the Natural Science Foundation of Chongqing under Grant No. cstc2019jcyj-msxmX0149 and the Project of Youth Innovation Promotion Association CAS.

References

Wenliang Chen, Yujie Zhang, and Hitoshi Isahara. 2006. An Empirical Study of Chinese Chunking. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 97–104, Sydney, Australia.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of

Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Shizhe Diao, Jiaxin Bai, Yan Song, Tong Zhang, and Yonggang Wang. 2019. ZEN: Pre-training Chinese Text Encoder Enhanced by N-gram Representations. *ArXiv*, abs/1911.00720.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning Word Vectors for 157 Languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan.

Honglei Guo, Huijia Zhu, Zhili Guo, Xiaoxun Zhang, Xian Wu, and Zhong Su. 2009. Domain Adaptation with Latent Semantic Association for Named Entity Recognition. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 281–289, Boulder, Colorado.

Binxuan Huang and Kathleen M Carley. 2019. Syntax-Aware Aspect Level Sentiment Classification with Graph Attention Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5472–5480.

Zhongqiang Huang, Mary Harper, and Wen Wang. 2007. Mandarin Part-of-Speech Tagging and Discriminative Reranking. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1093–1102, Prague, Czech Republic.

Wenbin Jiang, Liang Huang, and Qun Liu. 2009. Automatic Adaptation of Annotation Standards: Chinese Word Segmentation and POS Tagging – A Case Study. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 522–530, Suntec, Singapore.

Wenbin Jiang, Liang Huang, Qun Liu, and Yajuan Lü. 2008. A Cascaded Linear Model for Joint Chinese Word Segmentation and Part-of-Speech Tagging. In *Proceedings of ACL-08: HLT*, pages 897–904, Columbus, Ohio.

Nikita Kitaev and Dan Klein. 2018. Constituency Parsing with a Self-Attentive Encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia.

Canasai Kruengkrai, Kiyotaka Uchimoto, Jun'ichi Kazama, Yiou Wang, Kentaro Torisawa, and Hitoshi Isahara. 2009. An Error-Driven Word-Character Hybrid Model for Joint Chinese Word Segmentation and POS Tagging. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 513–521, Suntec, Singapore.

Abhishek Kumar, Daisuke Kawahara, and Sadao Kurohashi. 2018. Knowledge-Enriched Two-Layered Attention Network for Sentiment Analysis. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 253–258, New Orleans, Louisiana.

Shuhei Kurita, Daisuke Kawahara, and Sadao Kurohashi. 2017. Neural Joint Model for Transition-based Chinese Syntactic Analysis. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1204–1214, Vancouver, Canada.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland.

Katerina Margatina, Christos Baziotis, and Alexandros Potamianos. 2019. Attention-based Conditioning Methods for External Knowledge Integration. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3944–3951, Florence, Italy.

David McClosky, Eugene Charniak, and Mark Johnson. 2010. Automatic Domain Adaptation for Parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 28–36, Los Angeles, California.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-Value Memory Networks for Directly Reading Documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1409.

Hwee Tou Ng and Jin Kiat Low. 2004. Chinese Part-of-Speech Tagging: One-at-a-Time or All-at-Once? Word-Based or Character-Based? In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 277–284, Barcelona, Spain.

Joakim Nivre, Marie-Catherine De Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo,

Natalia Silveira, et al. 2016. Universal Dependencies v1: A Multilingual Treebank Collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 1659–1666.

Xian Qian and Yang Liu. 2012. Joint Chinese Word Segmentation, POS Tagging and Parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 501–511, Jeju Island, Korea.

Colin Raffel and Daniel PW Ellis. 2015. Feed-Forward Networks with Attention Can Solve Some Long-Term Memory Problems. *arXiv preprint arXiv:1512.08756*.

Dominic Seyler, Tatiana Dembelova, Luciano Del Corro, Johannes Hoffart, and Gerhard Weikum. 2018. A Study of the Importance of External Knowledge in the Named Entity Recognition Task. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 241–246.

Yan Shao, Christian Hardmeier, Jörg Tiedemann, and Joakim Nivre. 2017. Character-based Joint Segmentation and POS Tagging for Chinese using Bidirectional RNN-CRF. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 173–183, Taipei, Taiwan.

Mo Shen, Hongxiao Liu, Daisuke Kawahara, and Sadao Kurohashi. 2014. Chinese Morphological Analysis with Character-level POS Tagging. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 253–258, Baltimore, Maryland.

Yan Song, Chia-Jung Lee, and Fei Xia. 2017. Learning Word Representations with Regularization from Prior Knowledge. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 143–152, Vancouver, Canada.

Yan Song, Shuming Shi, Jing Li, and Haisong Zhang. 2018. Directional Skip-Gram: Explicitly Distinguishing Left and Right Context for Word Embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2*, pages 175–180, New Orleans, Louisiana.

Yan Song and Fei Xia. 2013. A Common Case of Jekyll and Hyde: The Synergistic Effect of Using Divided Source Training Data for Feature Augmentation. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 623–631, Nagoya, Japan.

Weiwei Sun. 2011. A Stacked Sub-Word Model for Joint Chinese Word Segmentation and Part-of-Speech Tagging. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1385–1394, Portland, Oregon, USA.

Yuanhe Tian, Yan Song, Fei Xia, Tong Zhang, and Yonggang Wang. 2020. Improving Chinese Word Segmentation with Wordhood Memory Networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Seattle, Washington, USA.

Yiou Wang, Jun'ichi Kazama, Yoshimasa Tsuruoka, Wenliang Chen, Yujie Zhang, and Kentaro Torisawa. 2011. Improving Chinese Word Segmentation and POS Tagging with Semi-supervised Methods Using Large Auto-Analyzed Data. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 309–317, Chiang Mai, Thailand.

Naiwen Xue, Fei Xia, Fu-Dong Chiou, and Marta Palmer. 2005. The Penn Chinese TreeBank: Phrase structure annotation of a large corpus. *Natural language engineering*, 11(2):207–238.

Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. 2020. Wikipedia2Vec: An Efficient Toolkit for Learning and Visualizing the Embeddings of Words and Entities from Wikipedia. *arXiv preprint 1812.06280v3*.

Xiaodong Zeng, Derek F. Wong, Lidia S. Chao, and Isabel Trancoso. 2013. Graph-based Semi-Supervised Model for Joint Chinese Word Segmentation and Part-of-Speech Tagging. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 770–779, Sofia, Bulgaria.

Hongming Zhang, Jiaxin Bai, Yan Song, Kun Xu, Changlong Yu, Yangqiu Song, Wilfred Ng, and Dong Yu. 2019. Multiplex Word Embeddings for Selectional Preference Acquisition. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5247–5256, Hong Kong, China.

Meishan Zhang, Nan Yu, and Guohong Fu. 2018. A Simple and Effective Neural Model for Joint Word Segmentation and POS Tagging. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 26(9):1528–1538.

Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. 2013. Deep Learning for Chinese Word Segmentation and POS Tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 647–657, Seattle, Washington, USA.