Word-Context Character Embeddings for Chinese Word Segmentation

Hao Zhou[*]
Nanjing University
& Toutiao AI Lab
zhouhao.nlp@bytedance.com

Zhenting Yu[*]
Nanjing University
yuzt@nlp.nju.edu.cn

Yue Zhang
Singapore University
of Technology and Design
yue_zhang@sutd.edu.sg

Shujian Huang
Nanjing University
huangsj@nlp.nju.edu.cn

Xinyu Dai
Nanjing University
daixinyu@nju.edu.cn

Jiajun Chen
Nanjing University
chenjj@nlp.nju.edu.cn

Abstract

Neural parsers have benefited from automatically labeled data via dependency-context word embeddings. We investigate training character embeddings on a word-based context in a similar way, showing that the simple method significantly improves state-of-the-art neural word segmentation models, beating tri-training baselines for leveraging auto-segmented data.

1 Introduction

Neural network Chinese word segmentation (CWS) models (Zhang et al., 2016; Liu et al., 2016; Cai and Zhao, 2016) appeal for their strong ability of feature representation, employing unigram and bigram character embeddings as input features (Zheng et al., 2013; Pei et al., 2014; Ma and Hinrichs, 2015; Chen et al., 2015a). They give state-of-the-art performances. We investigate leveraging automatically segmented texts for enhancing their accuracies.

Such semi-supervised methods can be divided into two main categories. The first one is *bootstrapping*, which includes *self-training* and *tri-training*. The idea is to generate more training instances by automatically labeling large-scale data. Self-training (Yarowsky, 1995; McClosky et al., 2006; Huang et al., 2010; Liu and Zhang, 2012) labels additional data by using the base classifier itself, and tri-training (Zhou and Li, 2005; Li et al., 2014) uses two extra classifiers, taking the instances with the same labels for additional training data. A second semi-supervised learning method in NLP is *knowledge distillation*, which extracts knowledge from large-scale auto-labeled data as features.

[*]Equal contributions

Tri-training has been used in neural parsing, giving considerable improvements for both of dependency (Weiss et al., 2015) and constituent parsing (Vinyals et al., 2015; Choe and Charniak, 2016). Knowledge from auto-labeled data has also been used for parsing (Bansal et al., 2014; Melamud et al., 2016), where word embeddings are trained on automatic dependency tree context. Such knowledge has also been proved effective in conventional discrete CWS models, such as *label distribution information* (Wang et al., 2011; Zhang et al., 2013). However, it has not been investigated for neural CWS.

We propose *word-context character embeddings* (WCC), using segmentation label information in the pre-training of unigram and bigram character embeddings. The method packs the label distribution information into the embeddings, which could be regarded as a way for knowledge parameterization. Our idea follows Levy and Goldberg (2014), who use *dependency contexts* to train *word embeddings*. Additionally, motivated by co-training, we propose *multi-view word-context character embeddings* for cross-domain segmentation, which pre-trains two types of embedding for in-domain and out-of-domain data, respectively. In-domain embeddings are used for solving data sparseness, and out-of-domain embeddings are used for domain adaptation.

Our proposed model is simple, efficient and effective, giving average 1% accuracy improvement on in-domain data and 3.5% on out-of-domain data, respectively, significantly out-performing self-training and tri-training methods for leveraging auto-segmented data.

2 Baseline Segmentation Model

Chinese word segmentation can be regarded as a character sequence labeling task, where each char-

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 769–775
Copenhagen, Denmark, September 7–11, 2017.

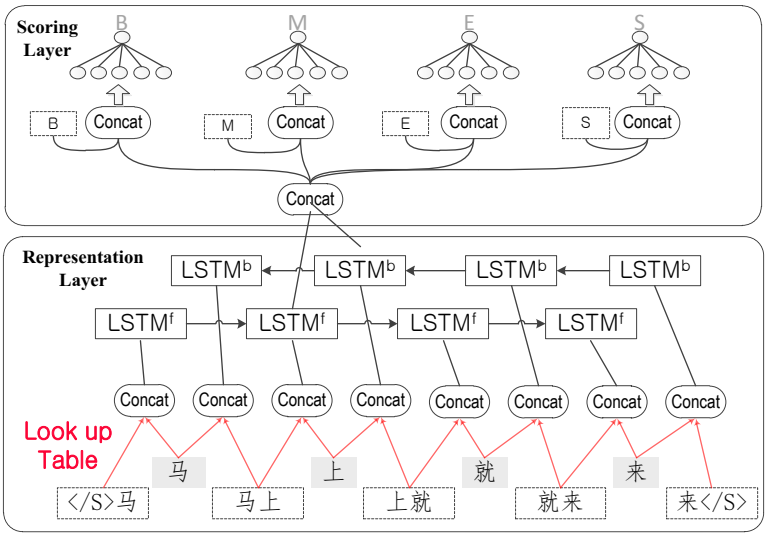

Figure 1: Baseline model architecture.

acter in the sentence is assigned a *segment label* from left to right, including {*B, M, E, S*}, to indicate the segmentation (Xue, 2003; Low et al., 2005; Zhao et al., 2006). *B, M, E* represent the character is the beginning, middle or end of a multi-character word, respectively. *S* represents that the current character is a single character word.

Following Chen et al. (2015b), a standard bi-LSTM model (Graves, 2008) is used to assign segmentation label for each character. As shown in Figure 1, our model consists of a *representation layer* and a *scoring layer*. The representation layer utilizes a bi-LSTM to capture the context of each character in the sentence. Given a sentence $\{w_1, w_2, w_3, \cdots, w_N\}$, where w_i is the i_{th} character in the sentence, and N is the sentence length, we have a corresponding embedding e_{w_i} and $e_{w_{i-1}w_i}$ for each character unigram w_i and character bigram $w_{i-1}w_i$, respectively. A forward word representation e_i^f is calculated as follows:

$$\begin{aligned} e_i^f &= concat_1(e_{w_i}, e_{w_{i-1}w_i}), \\ &= \tanh(W_1[e_{w_i}; e_{w_{i-1}w_i}]) \end{aligned}$$

A backward representation e_i^b can be obtained in the same way. Then e_i^f and e_i^b are fed into forward and backward LSTM units at current position, obtaining the corresponding forward and backward LSTM representations r_i^{lstm-f} and r_i^{lstm-b}, respectively.

In the *scoring layer*, we first obtain a linear combination of r_i^{lstm-f} and r_i^{lstm-b}, which is the final representation at the i_{th} position.

$$\begin{aligned} r_i &= concat_2(r_i^{lstm-f}, r_i^{lstm-b}) \\ &= \tanh(W_2[r_i^{lstm-f}; r_i^{lstm-b}]) \end{aligned}$$

Given the representation r_i, we use a scoring unit to score for each potential segment label. Given r_i, the score of segment label M is:

$$f_M^i = W_M h,$$

where

$$\begin{aligned} h &= concat_3(r_i, e_M), \\ &= \tanh(W_3[r_i; e_M]) \end{aligned}$$

W_M is the score matrix for label M, and e_M is the label embedding for label M.

3 Word-Context Character Embeddings

Our model structure is a derivation from the skip-gram model (Mikolov et al., 2013), similar to Levy and Goldberg (2014). Given a sentence with length n: $\{w_1, w_2, w_3, \cdots w_n\}$ and its corresponding segment labels: $\{l_1, l_2, l_3, \cdots l_n\}$, the pre-training context of current character w_t is the around characters in the windows with size c, together with their corresponding segment labels (Figure 2). Characters w_i and labels l_i in the context are represented by vectors $e_{w_i}^c \in \mathbb{R}^d$ and $e_{l_i}^c \in \mathbb{R}^d$, respectively, where d is the embedding dimensionality.

The word-context embedding of character w_t is represented as $e_{w_t} \in \mathbb{R}^d$, which is trained by predicting the surrounding context representations $e_{w'}^c$

and $e^c_{l_i}$, parameterizing the labeled segmentation information in the embedding parameters. To capture order information (Ling et al., 2015), we use different embedding matrices for context embedding in different context positions, training different embeddings for the same word when they reside on different locations as the context word. In particular, our context window size is five. As a result, each word has four different versions of e^c, namely e^c_{-1}, e^c_{-2}, e^c_{+1}, and e^c_{+2}, each taking a distinct embedding matrix. Given the context window $[w_{-2}, w_{-1}, w, w_{+1}, w_{+2}]$, w_{-1} is the left first context word of the focus word w, e^c_{-1,w_i} will be selected from embedding matrix E_{-1}, and w_{+1} is the right first word of w, e^c_{+1,w_i} will be selected from embedding matrix E_{+1}.

Note that each character has two types of embeddings, where e_{w_i} is the embedding form of w_i when w_i is the focus word, and $e^c_{w_i}$ is the embedding form of w_i when w_i is used as a surrounding context word. We do not have e_{l_i} because l_i only acts as the surrounding context. After pre-training, e_{w_i} will be used as the WCC embeddings.

The objective of our model is to maximize the average log probability of the context:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\le j\le c, j\neq 0} \log p(w_{t+j}|w_t) + \log p(t_{t+j}|w_t)$$

Negative sampling (Mikolov et al., 2013) is used, where $\log p(w_{t+j}|w_t)$ and $\log p(t_{t+j}|w_t)$ are computed as:

$$p(w_{t+j}|w_t) = \log\sigma({e^c_{w_{t+j}}}^\top e_{w_t}) + \sum_{i=1}^{k}\mathbb{E}_{w_i\sim P_n(w)}[\log\sigma(-{e^c_{w_i}}^\top e_{w_t})]$$

and

$$p(t_{t+j}|w_t) = \log\sigma({e^c_{l_{t+j}}}^\top e_{w_t}) + \sum_{i=1}^{k}\mathbb{E}_{l_i\sim P_n(l)}[\log\sigma(-{e^c_{l_i}}^\top e_{w_t})],$$

respectively, where $P_n(w)$ and $P_n(l)$ is the noise distributions and k is the size of negative samples for each data sample.

Bigram embeddings are trained in the same way as unigram character embeddings. For out-of-domain segmentation, we pre-train two embeddings for each token, extracting knowledge from the two domains, respectively.

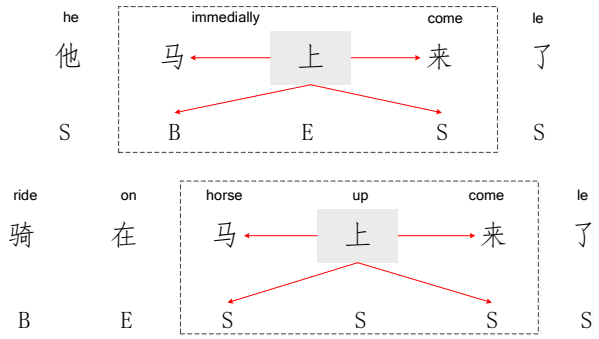

Figure 2: Word-context for the character ' 上' in two different sentences. The windows size $c = 3$.

4 Experiments

4.1 Set-up

We perform experiments on three standard datasets for Chinese word segmentation: PKU and MSR from the second SIGHAN bakeoff shared task, and Chinese Treebank 6.0 (CTB6). For PKU and MSR, 10% of the training data are randomly selected as development data. We follow Zhang et al. (2016) to split the CTB6 corpus into training, development and testing sections. For evaluating cross-domain performance, we also experiment on Chinese novel data. Following Zhang et al. (2014), the training set of CTB5 is selected for training, and the manually annotated sentences of free Internet novel 'Zhuxian' (ZX) are selected as the development and test data (Liu and Zhang, 2012)[1].

Chinese Gigaword (LDC2011T13, 4M) is used for in-domain unlabeled data. For out-of-domain data, 20K raw sentences of Zhuxian is used. We take self-training and tri-training as baselines, which also use large-scale auto-segmented data. For self-training, skip-gram pre-training and word-context character embedding, unlabeled corpus is segmented automatically by our baseline model. For tri-training, we additionally use the ZPar (Zhang and Clark, 2007) and ICTCLAS[2] as our base classifiers .

We use F1 to evaluate segmentation accuracy. The recalls of in-vocabulary (IV) and out-of-vocabulary (OOV) are also measured.

4.2 Hyper-Parameters

The hyper-parameters used in this work are listed in Table 1. The values are selected according

[1] http://zhangmeishan.github.io/eacl14mszhang.zip

[2] http://ictclas.nlpir.org/

unigram dimension	50
bigram dimesion	50
label embedding dimention	32
LSTM hidden size	100
LSTM input size	100
learning rate	0.1
windows size	5

Table 1: Hyper-parameters.

System	CTB6	PKU	MSR	Speed
Greedy	94.9	95.0	97.2	14.7
CRF	95.0	95.1	97.2	3.6

Table 2: Comparisons between greedy and CRF segmentation. Speed: tokens per millisecond.

to the development set of CTB6. Many previous character-based CWS models use a transition matrix to model the tag dependency and CRF for structured inference (Pei et al., 2014; Chen et al., 2015a). However, we find that, the greedy model obtains comparable segmentation accuracies across CTB6, PKU and MSR, yet giving much fast speed (Table 2). Hence we adopt the greedy model as our baseline segmentation model.

4.3 Utilizing Varying-Scale Data

The results of self-training and tri-training with varying-scale training data are list in Table 3, where +4X means adding 4 times the size of supervised training data into the training set. We find that self-training does not work well, and tri-training with 16X gives a 0.5% accuracy improvement. We adopt this setting for our baseline in the remaining experiments[3].

We also try to choose more effective examples for self-training and tri-training, by selecting training instances according to the base segmentation model score. However, the segmentation performances do not get improved. A possible reason is that the training instances with higher confidence are always shorter than the original sampled sentences, which may not be very helpful for semi-spervised segmentation.

4.4 In-Domain Results

As shown in Table 4, pre-training with conventional skip-gram embeddings gives only small improvements, which is consistent as findings of previous work (Chen et al., 2015a; Ma and Hinrichs, 2015; Cai and Zhao, 2016). Segmentation with self-training even shows accuracy drops on PKU and MSR. We speculate that the self-training by the neural CWS baseline is sensitive to the segmentation errors of the auto-labeled data. On average, our method obtains an absolute 1% accuracy improvement over the baseline, outperforming other semi-supervised method significantly[4].

We compare our model with other state-of-the-art segmentation models[5], which are grouped into 3 classes, namely traditional segmentation models (*non-nn*), neural segmentation models (*nn*), and the combination of both neural and traditional discrete features (*comb*). Our simple model gives top accuracies compared with related work. Liu et al. (2016), Cai and Zhao (2016) and Zhang et al. (2016) propose to incorporate word embedding features in the neural CWS, pre-training the word embeddings in the large-scale labeled data. Different to them, we employ a simpler character level model containing word information, yet obtaining higher F1 scores.

Systems	+4X	+8X	+16X	+32X
baseline	94.9			
self-training	95.0	94.9	94.9	94.8
tri-training	95.2	95.3	95.4	95.4

Table 3: Results of self-training and tri-training on CTB6 with varying scaled training data.

Type	System	CTB6	PKU	MSR
non-nn	Tseng et al. (2005)	-	95.0	96.4
	Sun et al. (2009)	-	95.2	97.3
	Wang et al. (2011)	95.8	-	-
	Zhang et al. (2013)	-	**96.1**	97.5
nn	Zheng et al. (2013)	-	92.4	93.3
	Pei et al. (2014)	-	95.2	97.2
	Kong et al. (2015)	-	90.6	90.7
	Ma and Hinrichs (2015)	-	95.1	96.6
	Chen et al. (2015c)†	-	94.8	95.6
	Xu and Sun (2016)	95.8	**96.1**	96.3
	Liu et al. (2016)	95.5	95.7	97.6
	Zhang et al. (2016)	95.4	95.1	97.0
	Cai and Zhao (2016)	-	95.5	96.5
comb	Zhang et al. (2016)	96.0	95.7	97.7
Ours	*baseline*	94.9	95.0	97.2
	+ *self-training*	95.0	94.8	97.0
	+ *tri-training*	95.5	95.5	97.4
	+ *skip-gram embeddings*	95.3	95.5	97.4
	+ *WCC embeddings*	**96.2**	**96.0**	**97.8**

Table 4: Comparison with other models.

[3]For out-of-domain experiments, we include both the +16X and the 20K out-of-domain data for self-training and tri-training.

[4]The p-values are below 0.01 using pairwise t-test.

[5]Results with † are obtained from Cai and Zhao (2016), because results in the original paper use dictionary resources.

ours:	若在 (if)_ 鬼王 (guiwang)_ 手上 (hand) 夺下 (wrest)_ 七星剑 (qixin sword)，_ 我 (I) 必 (must)_ 器重 (think highly of)_ 于 (at) 你 (you)
baseline:	若在 (if)_ 鬼 (gui)_ 王 (king)_ 手上 (hand) 夺下 (wrest)_ 七 (seven)_ 星 (star)_ 剑 (sword)， 我 (I)_ 必 (must)_ 器 (ware)_ 重 (heavy) 于 (at)_ 你 (you)

Figure 3: Case studies.

System		F1	ROOV	RIV
Zhang et al. (2014b))	*baseline*	87.7	-	-
	+ *self-training*	88.7	-	-
Liu et al. (2014)	*baseline*	87.5	-	-
	+*Chinese Wikipedia*	**90.6**	-	-
Ours	*baseline*†	86.6	60.8	91.7
	+ *skip-gram*‡	87.6	-	-
	+ *self-training*‡	87.8	70.3	91.5
	+ *tri-training*‡	88.1	68.1	91.5
	+ *WCC embeddings*‡	89.1	70.4	93.7
	+ *multi-view WCC embeddings*♯	**90.1**	**74.1**	**93.3**

Table 5: Results on the out-of-domain data. Models with † do not use large-scale data, models with ‡ use in-domain large-scale data, and models with ♯ use both in-domain, and out-of-domain large-scale data.

4.5 Out-of-Domain Results

We test out-of-domain performance of our model on the ZX dataset. We also use the multi-view word-context character embeddings (WCC) for cross domain segmentation, which uses two types of embeddings by simple vector concatenation. One type of embeddings is pre-trained on in-domain data, and the other type is pre-trained on out-of-domain data. In such case, the multi-view embeddings includes cross-domain information, which may enhance the cross-domain segmentation performance (Mou et al., 2016).

As shown in Table 5, using word-context character (*WCC*) embeddings and multi-view word-context character embeddings both give significantly higher accuracy improvements compared with other semi-supervised methods. Additionally, we find that multi-view WCC embeddings give an extra 1% F1 score improvement over WCC embeddings. Our proposed model also significantly improves the OOV recall (ROOV) and IV recall (RIV). By studying the cases of segmented output (Figure 3), we find that our model can recognize OOV words such as '鬼王', '七星剑' and the IV word '器重', which are incorrectly labeled by the baseline. This confirms that our proposed model is helpful for the data sparseness problem on closed domain and domain adaptation on across domain.

We also list the results of Zhang et al. (2014) and Liu et al. (2014) on this dataset. Liu et al. (2014) obtains better out-of-domain performance than our model. However, their results cannot be compared directly with ours because they use partial labeled URL link data from Chinese Wikipedia data for training.

5 Conclusion

We proposed word-context character embeddings for semi-supervised neural CWS, which makes the segmentation model more accurate on in-domain data, and more robust on the out-of-domain data. Our segmentation model is simple yet effective, achieving state-of-the-art segmentation accuracies on standard benchmarks. It can also be useful for other NLP tasks with small labeled training data, but a large unlabeled data. Our code could be downloaded at `https://github.com/zhouh/WCC-Segmentation`.

Acknowledge

We would like to thank the anonymous reviewers for their insightful comments. We also thank Ji Ma and Meishan Zhang for their helpful discussions. This work was partially founded by the Natural Science Foundation of China (61672277, 71503124) and the China National 973 project 2014CB340301.

References

Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring continuous word representations for dependency parsing. In *ACL (2)*, pages 809–815.

Deng Cai and Hai Zhao. 2016. Neural word segmentation learning for chinese. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 409–420. Association for Computational Linguistics.

Wenliang Chen, Min Zhang, and Yue Zhang. 2013. Semi-supervised feature transformation for dependency parsing. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1303–1313. Association for Computational Linguistics.

Wenliang Chen, Yue Zhang, and Min Zhang. 2014. Feature embedding for dependency parsing. In *COLING*, pages 816–826.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, and Xuanjing Huang. 2015a. Gated recursive neural network for chinese word segmentation. In *Proceedings of Annual Meeting of the Association for Computational Linguistics*.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. 2015b. Long short-term memory neural networks for chinese word segmentation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. 2015c. Long short-term memory neural networks for chinese word segmentation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1197–1206. Association for Computational Linguistics.

Kook Do Choe and Eugene Charniak. 2016. Parsing as language modeling. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2331–2336. Association for Computational Linguistics.

Alex Graves. 2008. *Supervised Sequence Labelling with Recurrent Neural Networks*. Ph.D. thesis, Technical University Munich.

Zhongqiang Huang, Mary Harper, and Slav Petrov. 2010. Self-training with products of latent variable grammars. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 12–22. Association for Computational Linguistics.

Lingpeng Kong, Chris Dyer, and Noah A Smith. 2015. Segmental recurrent neural networks. *ICLR*.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *ACL (2)*, pages 302–308. Citeseer.

Zhenghua Li, Min Zhang, and Wenliang Chen. 2014. Ambiguity-aware ensemble training for semi-supervised dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 457–467. Association for Computational Linguistics.

Wang Ling, Chris Dyer, Alan Black, and Isabel Trancoso. 2015. Two/too simple adaptations of word2vec for syntax problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

Yang Liu and Yue Zhang. 2012. Unsupervised domain adaptation for joint segmentation and pos-tagging. In *COLING (Posters)*, pages 745–754.

Yijia Liu, Wanxiang Che, Jiang Guo, Bing Qin, and Ting Liu. 2016. Exploring segment representations for neural segmentation models. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*.

Yijia Liu, Yue Zhang, Wanxiang Che, Ting Liu, and Fan Wu. 2014. Domain adaptation for crf-based chinese word segmentation using free annotations. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 864–874. Association for Computational Linguistics.

Jin Kiat Low, Hwee Tou Ng, and Wenyuan Guo. 2005. A maximum entropy approach to chinese word segmentation. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, volume 1612164, pages 448–455.

Jianqiang Ma and Erhard Hinrichs. 2015. Accurate linear-time chinese word segmentation via embedding matching. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1733–1743. Association for Computational Linguistics.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the main conference on human language technology conference of the North American Chapter of the Association of Computational Linguistics*, pages 152–159. Association for Computational Linguistics.

Oren Melamud, David McClosky, Siddharth Patwardhan, and Mohit Bansal. 2016. The role of context types and dimensionality in learning word embeddings. In *Proceedings of NAACL-HLT*, pages 1030–1040.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. 2016. How transferable are neural networks in nlp applications? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 479–489, Austin, Texas. Association for Computational Linguistics.

Wenzhe Pei, Tao Ge, and Baobao Chang. 2014. Max-margin tensor neural network for chinese word segmentation. In *ACL (1)*, pages 293–303.

Xu Sun, Yaozhong Zhang, Takuya Matsuzaki, Yoshimasa Tsuruoka, and Jun'ichi Tsujii. 2009. A discriminative latent variable chinese segmenter with hybrid word/character information. In *Proceedings of Human Language Technologies: The 2009 Annual*

Conference of the North American Chapter of the Association for Computational Linguistics, pages 56–64. Association for Computational Linguistics.

Huihsin Tseng, Pichuan Chang, Galen Andrew, Daniel Jurafsky, and Christopher Manning. 2005. A conditional random field word segmenter for sighan bakeoff 2005. In *Proceedings of the fourth SIGHAN workshop on Chinese language Processing*, volume 171.

Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey E. Hinton. 2015. Grammar as a foreign language. In *NIPS*.

Yiou Wang, Yoshimasa Tsuruoka 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 *IJCNLP*, pages 309–317.

David Weiss, Chris Alberti, Michael Collins, and Slav Petrov. 2015. Structured training for neural network transition-based parsing. *arXiv preprint arXiv:1506.06158*.

Jingjing Xu and Xu Sun. 2016. Dependency-based gated recursive neural network for chinese word segmentation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 567–572. Association for Computational Linguistics.

Nianwen Xue. 2003. Chinese word segmentation as character tagging. *Computational Linguistics and Chinese Language Processing*, 8(1):29–48.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196. Association for Computational Linguistics.

Longkai Zhang, Houfeng Wang, Xu Sun, and Mairgup Mansur. 2013. Exploring representations from unlabeled data with co-training for chinese word segmentation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 311–321. Association for Computational Linguistics.

Meishan Zhang, Yue Zhang, Wanxiang Che, and Ting Liu. 2014. Type-supervised domain adaptation for joint segmentation and pos-tagging. In *EACL*, pages 588–597.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2016. Transition-based neural word segmentation. In *Proceedings of the 54nd Annual Meeting of the Association for Computational Linguistics*.

Yue Zhang and Stephen Clark. 2007. Chinese segmentation with a word-based perceptron algorithm. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 840–847, Prague, Czech Republic. Association for Computational Linguistics.

Hai Zhao, Chang-Ning Huang, and Mu Li. 2006. An improved chinese word segmentation system with conditional random field. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, volume 1082117. Sydney: July.

Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. 2013. Deep learning for chinese word segmentation and pos tagging. In *EMNLP*, pages 647–657.

Zhi-Hua Zhou and Ming Li. 2005. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Transactions on knowledge and Data Engineering*, 17(11):1529–1541.

Muhua Zhu, Yue Zhang, Wenliang Chen, Min Zhang, and Jingbo Zhu. 2013. Fast and accurate shift-reduce constituent parsing. In *51st Annual Meeting of the Association for Computational Linguistics*.