Fast and Accurate Neural Word Segmentation for Chinese

Deng Cai[1,2], **Hai Zhao**[1,2,*], **Zhisong Zhang**[1,2], **Yuan Xin**[3], **Yongjian Wu**[3], **Feiyue Huang**[3]
[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2]Key Lab of Shanghai Education Commission for Intelligent Interaction
and Cognitive Engineering, Shanghai Jiao Tong Univeristy, Shanghai, China
thisisjcykcd@gmail.com, {zhaohai@cs, zzs2011@}sjtu.edu.cn
[3]Tencent Youtu Lab, Shanghai, China
{macxin,littlekenwu,garyhuang}@tencent.com

Abstract

Neural models with minimal feature engineering have achieved competitive performance against traditional methods for the task of Chinese word segmentation. However, both training and working procedures of the current neural models are computationally inefficient. This paper presents a greedy neural word segmenter with balanced word and character embedding inputs to alleviate the existing drawbacks. Our segmenter is truly end-to-end, capable of performing segmentation much faster and even more accurate than state-of-the-art neural models on Chinese benchmark datasets.

1 Introduction

Word segmentation is a fundamental task for processing most east Asian languages, typically Chinese. Almost all practical Chinese processing applications essentially rely on Chinese word segmentation (CWS), e.g., (Zhao et al., 2017).

Since (Xue, 2003), most methods formalize this task as a sequence labeling problem. In a supervised learning fashion, sequence labeling may adopt various models such as Maximum Entropy (ME) (Low et al., 2005) and Conditional Random Fields (CRF) (Lafferty et al., 2001; Peng et al., 2004). However, these models rely heavily on hand-crafted features.

*Corresponding author. This paper was partially supported by Cai Yuanpei Program (CSC No. 201304490199 and No. 201304490171), National Natural Science Foundation of China (No. 61170114, No. 61672343 and No. 61272248), National Basic Research Program of China (No. 2013CB329401), Major Basic Research Program of Shanghai Science and Technology Committee (No. 15JC1400103), Art and Science Interdisciplinary Funds of Shanghai Jiao Tong University (No. 14JCRZ04), Key Project of National Society Science Foundation of China (No. 15-ZDA041), and the joint research project with Youtu Lab of Tencent.

To minimize the efforts in feature engineering, neural word segmentation has been actively studied recently. Zheng et al. (2013) first adapted the sliding-window based sequence labeling (Collobert et al., 2011) with character embeddings as input. A number of other researchers have attempted to improve the segmenter of (Zheng et al., 2013) by augmenting it with additional complexity. Pei et al. (2014) introduced tag embeddings. Chen et al. (2015a) proposed to model n-gram features via a gated recursive neural network (GRNN). Chen et al. (2015b) used a Long short-term memory network (LSTM) (Hochreiter and Schmidhuber, 1997) to capture long-distance context. Xu and Sun (2016) integrated both GRNN and LSTM for deeper feature extraction.

Besides sequence labeling schemes, Zhang et al. (2016) proposed a transition-based framework. Liu et al. (2016) used a zero-order semi-CRF based model. However, these two models rely on either traditional discrete features or non-neural-network components for performance enhancement, their performance drops rapidly when solely depending on neural models. Most closely related to this work, Cai and Zhao (2016) proposed to score candidate segmented outputs directly, employing a gated combination neural network over characters for word representation generation and an LSTM scoring model for segmentation result evaluation.

Despite the active progress of most existing works in terms of accuracy, their computational needs have been significantly increased to the extent that training a neural segmenter usually takes days even using cutting-edge hardwares. Meanwhile, different applications often require diverse segmenters and offer large-scale incoming data. The efficiency of a word segmenter either for training and decoding is crucial in practice. In this paper, we propose a simple yet accurate neu-

Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 608–615
Vancouver, Canada, July 30 - August 4, 2017.
https://doi.org/10.18653/v1/P17-2096

ral word segmenter who searches greedily during both training and working to overcome the existing efficiency obstacle. Our evaluation will be performed on Chinese benchmark datasets.

2 Related Work

Statistical Chinese word segmentation has been studied for decades (Huang and Zhao, 2007). (Xue, 2003) was the first to cast it as a character-based tagging problem. Peng et al. (2004) showed CRF based model is particularly effective to solve CWS in the sequence labeling fashion. This method has been followed by most later segmenters (Tseng et al., 2005; Zhao et al., 2006; Zhao and Kit, 2008c; Zhao et al., 2010; Sun et al., 2012; Zhang et al., 2013). The same spirit has also be followed by most neural models (Zheng et al., 2013; Pei et al., 2014; Qi et al., 2014; Chen et al., 2015a,b; Ma and Hinrichs, 2015; Xu and Sun, 2016).

Word based CWS to conveniently incorporate complete word features has also be explored. Andrew (2006) proposed a semi-CRF model. Zhang and Clark (2007, 2011) used a perceptron algorithm with inexact search. Both of them have been followed by neural model versions (Liu et al., 2016) and (Zhang et al., 2016) respectively. There are also works integrating both character-based and word-based segmenters (Huang and Zhao, 2006; Sun, 2010; Wang et al., 2014) and semi-supervised learning (Zhao and Kit, 2008b, 2011; Zeng et al., 2013; Zhang et al., 2013).

Unlike most previous works, which extract features within a fixed sized sliding window, Cai and Zhao (2016) proposed a direct segmentation framework that extends the feature window to cover complete input and segmentation history and uses beam search for decoding. In this work, we will make a series of significant improvement over the basic framework and especially adopt greedy search instead.

Another notable exception of embedding based methods is (Ma and Hinrichs, 2015), which used character-specified tags matching for fast decoding and resulted in a character-based greedy segmenter.

3 Models

To segment a character sequence, we employ neural networks to score the likelihood of a candidate segmented sequence being a true sentence, and the one with the highest score will be picked as output.

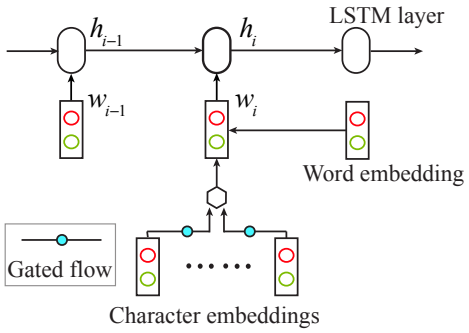

Figure 1: Neural network scoring for word candidate w_i in a possible word sequence $(..., w_i, ...)$.

3.1 Neural Scorer

Our neural architecture to score a segmented sequence (word sequence) can be described in the following three steps (illustrated in Figure 1).

Encoding To make use of neural networks, symbolic data needs to be transformed into distributed representations. The most straightforward solution is to use a lookup table for word vectors (Bengio et al., 2003). However, in the context of neural word segmentation, it will generalize poorly due to the severe word sparsity in Chinese. An alternative is employing neural networks to compose word representations from character embedding inputs. However, it is empirically hard to learn a satisfactory composition function. In fact, quite a lot of Chinese words, like "沙(sand)发(issue)" (sofa) , are not semantically character-level compositional at all.

For the dilemma that composing word representations from character may be insufficient while the direct use of word embedding may lose generalization ability, we propose a hybrid mechanism to alleviate the problem. Concretely, we keep a short list $\mathcal{H}$ of the most frequent words $w = c_1..c_l$ to balance character composition. If w in $\mathcal{H}$, the immediate word embedding $\mathbf{w} \in \mathbb{R}^{d_w}$ is attached via average pooling[1], otherwise, the character composition is used alone.

$$\textsc{Word}(c_1..c_l) = \begin{cases} \frac{\textsc{Comp}(c_1..c_l)+\mathbf{w}[w]}{2} & \text{if } c_1..c_l \in \mathcal{H} \\ \textsc{Comp}(c_1..c_l) & \text{otherwise} \end{cases}$$

Our character composition function $\textsc{Comp}(\cdot)$ for

[1]We tried other two integration functions, concatenation and adaptive gating mechanism, but it finally shows that the simplest averaging works best.

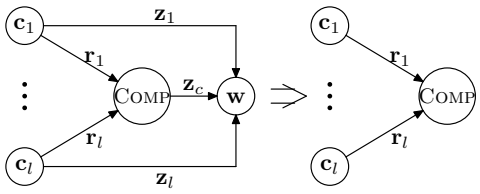

Figure 2: The difference between (Cai and Zhao, 2016) (left) and our model (right).

l-length word is

$$\textsc{Comp}(c_1..c_l) = \tanh(\mathbf{W}_l^c[\mathbf{r}_1 \odot \mathbf{c}_1; \ldots; \mathbf{r}_l \odot \mathbf{c}_l] + \mathbf{b}_l^c)$$

where $\odot$ denotes the element-wise multiplication. $\mathbf{r}_i \in \mathbb{R}^{d_c}$ is the gate that controls the information flow from character embedding $\mathbf{c}_i \in \mathbb{R}^{d_c}$ to word. Intuitively, the gating mechanism is used to determine which part of the character vectors should be retrieved when composing a certain word. This is indeed important due to the ambiguity of individual Chinese characters.

$$[\mathbf{r}_1; \ldots; \mathbf{r}_l] = \text{sigmoid}(\mathbf{W}_l^r[\mathbf{c}_1; \ldots; \mathbf{c}_l] + \mathbf{b}_l^r)$$

In contrast, the model in (Cai and Zhao, 2016) further combined $\textsc{Comp}(\cdot)$ and character embeddings $\mathbf{c}_i$ via an update gate $\mathbf{z}$ (As in Figure 2), which has been shown helpless to the performance but requires huge computational cost according to our empirical study.

Linking To capture word interactions within a word sequence, the resulted word vectors are then linked sequentially via an LSTM (Sundermeyer et al., 2012). At each time step i, a prediction about next word is made according to the current hidden state $\mathbf{h}_i \in \mathbb{R}^H$ of LSTM. The procedure can be described as the following equation.

$$\mathbf{p}_{i+1} = \tanh(\mathbf{W}^p\mathbf{h}_i + \mathbf{b}^p)$$

The predictions $\mathbf{p} \in \mathbb{R}^{d_w}$ will then be used to evaluate how reasonable the transition is between next word and the preceding word sequence.

Scoring The segmented sequence is evaluated from two perspectives, (i) the legality of individual words, (ii) the smoothness or coherence of the word sequence. The former is judged by a trainable parameter vector $\mathbf{u} \in \mathbb{R}^{d_w}$, which is supposed to work like a hyperplane separating legal and illegal words. For the latter, the prediction $\mathbf{p}$ made for each position can be used to score the fitness of the actual word. Both scoring operations are implemented via dot product in our settings. Summing up all scores, the segmented sequence (sentence) is scored as follow.

$$s([w_1, w_2, \ldots, w_n]) = \sum_{i=1}^{n} (\mathbf{u} + \mathbf{p}_i)^{\mathrm{T}} \textsc{Word}(w_i)$$

3.2 Search

The number of possible segmented sentences grows exponentially with the length of the input character sequence. Most existing methods made Markov assumptions to keep the exact search tractable.[2] However, such assumptions cannot be made in our model as the LSTM component takes advantage of the full segmentation history. We then adopt a beam search scheme, which works iteratively on every prefix of the input character sequence, approximating the k highest-scored word sequences of each prefix (i.e., k is the beam size). The time complexity of our beam search is $O(wkn)$, where w is the maximum word length and n is the input sequence length.

3.3 Training Criteria

Our segmenter is trained using max-margin methods (Taskar et al., 2005) where the structured margin loss is defined as μ times the number of incorrectly segmented characters (Cai and Zhao, 2016). However, according to (Huang et al., 2012), **standard** parameter update cannot guarantee convergence in the case of inexact search. We thus additionally examine two strategies as follows.

Early update This strategy proposed in (Collins and Roark, 2004) can be simplified into "update once the golden answer is unreachable". In our case, when the considering character prefix can be correctly segmented but the correct one falls off the beam, an update operation will be conducted and the rest part will be ignored.

LaSO update One drawback of early update is that the search may never reach the end of a training instance, which means the rest part of the instance is "wasted". Differently, LaSO method of (Daumé III and Marcu, 2005) continues on the same instance with correct hypothesis after each update. In our case, the beam will be emptied and the corresponding prefix of the correct word sequence will be inserted into the beam.

[2] By assuming that tag interactions or word interactions only exist in adjacent positions.

	PKU		MSR	
	Train	Test	Train	Test
#sentences	19K	2K	87K	4K
#words	1,110K	104K	2,368K	107K
#characters	1,788K	169K	3,981K	181K

Table 1: Data statistics.

Character embedding size	$d_c = 50$
Word embedding size	$d_w = 50$
Hidden unit number	$H = 50$
Margin loss discount	$\mu = 0.2$
Maximum word length	$w = 4$

Table 2: Model setting.

4 Experiments

4.1 Datasets and Settings

We conduct experiments on two popular benchmark datasets, namely PKU and MSR, from the second international Chinese word segmentation bakeoff (Emerson, 2005) (Bakeoff-2005). Data statistics are in Table 1.

Throughout this paper, we use the same model setting as shown in Table 2. These numbers are tuned on development sets.[3] We follow (Dyer et al., 2015) to train model parameters. The learning rate at epoch t is set as $\eta_t = 0.2/(1 + \gamma t)$, where $\gamma = 0.1$ for PKU dataset and $\gamma = 0.2$ for MSR dataset. The character embeddings are either randomly initialized or pre-trained by word2vec (Mikolov et al., 2013) toolkit on Chinese Wikipedia corpus (which will be indicated by *+pre-train* in tables.), while the word embeddings are always randomly initialized. The beam size is kept the same during training and working. By default, early update strategy is adopted and the word table H is top half of in-vocabulary (IV) words by frequency.

4.2 Model Analysis

Beam search collapses into greedy search Figure 3 demonstrates the effect of beam size. To our surprise, beam size change has little influence on the performance. Namely, simple stepwise greedy search nearly achieves the best performance, which suggests that word segmentation can be greedily solvable at word-level. It may be due to that right now the model is optimal enough to make correct decisions at the first position. In fact, similar phenomenon was observed at character-level (Ma and Hinrichs, 2015). The rest experiments will thus only report the results of our greedy segmenter.

[3]Following conventions, the last 10% sentences of training corpus are used as development set.

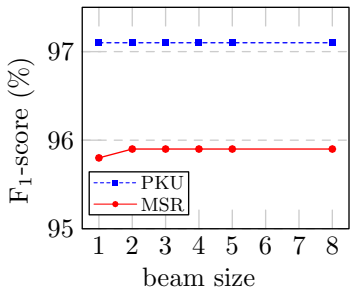

Figure 3: The effect of different beam sizes.

Methods	PKU		MSR	
	F_1	#epochs	F_1	#epochs
Standard	95.6	50	96.7	50
Early update	**95.8**	**30**	**97.1**	**30**
LaSO update	95.7	45	97.0	30

Table 3: The effect of different update methods. #epochs denotes the number of training epochs to convergence.

Comparing different update methods Table 3 compares the concerned three training strategies. We find that early update leads to faster convergence and a bit better performance compared to both standard and LaSO update.

Character composition versus word embedding Following Section 3.1, direct use of word embedding may bring efficiency and effectiveness for identifying IV words, but weaken the ability to recognize out-of-vocabulary (OOV) words. We accordingly conduct experiments on different sizes of word table $\mathcal{H}$. Concretely, sorting all IV words by frequency, the first {0, 25%, 50%, 75%, 100%} fraction of them respectively forms the word table. The corresponding results on PKU in Figure 4 demonstrate that by the use of direct word embedding, F_1 score increases first but then drops rapidly. In contrast, OOV recall, which partially reflects the model generalization ability, decreases consistently. In addition, we also found the number of training epochs to convergence decreases continually. Overall, the results indicate that word-aware segmenter learns faster and fits better on training set, but generalizes relatively poorly.

Models	PKU				MSR			
	F_1 + *pre-train*	F_1	Training (hours)	Test (sec.)	F_1 + *pre-train*	F_1	Training (hours)	Test (sec.)
(Zhao and Kit, 2008c)	-	95.4	-	-	-	**97.6**	-	-
(Chen et al., 2015a)	94.5*	94.4*	50	105	95.4*	95.1*	100	120
(Chen et al., 2015b)	94.8*	94.3*	58	105	95.6*	95.0*	117	120
(Ma and Hinrichs, 2015)	-	95.1	**1.5**	**24**	-	96.6	**3**	**28**
(Zhang et al., 2016)	95.1	-	6	110	97.0	-	13	125
(Liu et al., 2016)	93.91	-	-	-	95.21	-	-	-
(Cai and Zhao, 2016)	95.5	95.2	48	95	96.5	96.4	96	105
Our results	**95.8**	**95.4**	3	25	**97.1**	97.0	6	30

Table 4: Comparison with previous models. Results with * are from (Cai and Zhao, 2016).[4]

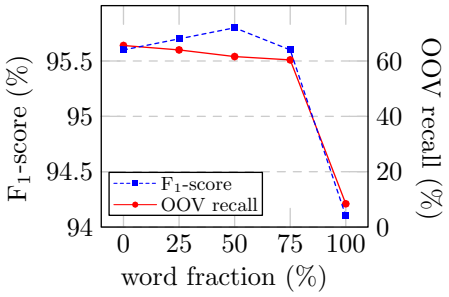

Figure 4: Performance with different sizes of word table on PKU test set.

4.3 Main Results

Table 4 compares our final results (greedy search is adopted by setting k=1) to prior neural models. Pre-training character embeddings on large-scale unlabeled corpus (not limited to the training corpus) has been shown helpful for extra performance improvement. The results with or without pre-trained character embeddings are listed separately for following the strict closed test setting of SIGHAN Bakeoff in which no linguistic resource other than training corpus is allowed. We also show the state-of-the-art results in (Zhao and Kit, 2008b) of traditional methods. The comparison shows our neural word segmenter outperforms all state-of-the-art neural systems with much less computational cost.

Finally, we present the results on all four Bakeoff-2005 datasets compared to (Zhao and Kit, 2008c) in Table 5. Note (Zhao and Kit, 2008c) used AV features, which are derived from global statistics over entire training corpus in a similar way of unsupervised segmentation (Zhao and Kit, 2008a), for performance enhancement.[5] The comparison to their results without AV features show that our neural models for the first time present comparable performance against state-of-the-art traditional ones under strict closed test setting.[6]

Models	PKU	MSR	CityU	AS
(Zhao and Kit, 2008c)	95.4	97.6	96.1	95.7
–AV	95.2	**97.4**	94.8	**95.3**
ours	**95.4**	97.0	**95.4**	95.2
+pre-train	95.8	97.1	95.6	95.3

Table 5: Results on all four Bakeoff-2005 datasets.

5 Conclusion

In this paper, we presented a fast and accurate word segmenter using neural networks. Our experiments show a significant improvement over existing state-of-the-art neural models by adopting the following key model refinements.

(1) A novel character-word balanced mechanism for word representation generation. (2) A more efficient model for character composition by dropping unnecessary designs. (3) Early update strategy during max-margin training. (4) With the above modifications, we discover that beam size has little influence on the performance. Actually, greedy search achieves very high accuracy.

Through these improvement from both neural models and linguistic motivation, our model becomes simpler, faster and more accurate.[7]

[4]To distinguish the performance improvement from model optimization, we especially list the results of standalone neural models in (Zhang et al., 2016) and (Liu et al., 2016). All the running time results are from our runs of released implementations on a single CPU (Intel i7-5960X) with two threads only, except for those of (Zhang et al., 2016) which are from personal communication. The results of (Xu and Sun, 2016) are not listed due to their use of external Chinese idiom dictionary.

[5]In fact, this kind of features may also be incorporated to our model. We leave it as future work.

[6]To our knowledge, none of previous neural models has ever performed a complete evaluation over all four segmentation corpora of Bakeoff-2005, in which only two, PKU and MSR, are used since (Pei et al., 2014).

[7]Our code based on Dynet (Neubig et al., 2017) is released at `https://github.com/jcyk/greedyCWS`.

References

Galen Andrew. 2006. A hybrid markov/semi-markov conditional random field for sequence segmentation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. Sydney, Australia, pages 465–472.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *The Journal of Machine Learning Research* 3:1137–1155.

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)*. Berlin, Germany, pages 409–420.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, and Xuanjing Huang. 2015a. Gated recursive neural network for Chinese word segmentation. 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)*. Beijing, China, pages 1744–1753.

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 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal, pages 1197–1206.

Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics, Main Volume*. Barcelona, Spain, pages 111–118.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research* 12:2493–2537.

Hal Daumé III and Daniel Marcu. 2005. Learning as search optimization: Approximate large margin methods for structured prediction. In *Proceedings of the 22nd international conference on Machine learning*. Bonn, Germany, pages 169–176.

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

Thomas Emerson. 2005. The second international Chinese word segmentation bakeoff. In *Proceedings of the fourth SIGHAN workshop on Chinese language Processing*. Jeju Island, Korea, pages 123–133.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Chang-Ning Huang and Hai Zhao. 2006. Which is essential for Chinese word segmentation: Character versus word. In *The 20th Pacific Asia Conference on Language, Information and Computation*. Wuhan, China, pages 1–12.

Changning Huang and Hai Zhao. 2007. Chinese word segmentation: A decade review. *Journal of Chinese Information Processing* 21(3):8–20.

Liang Huang, Suphan Fayong, and Yang Guo. 2012. Structured perceptron with inexact search. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Montréal, Canada, pages 142–151.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth Interntional Conference on Machine Learning*. San Francisco, USA, volume 1, pages 282–289.

Yijia Liu, Wanxiang Che, Jiang Guo, Bing Qin, and Ting Liu. 2016. Exploring segment representations for neural segmentation models. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. New York, USA, pages 2880–2886.

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*. Jeju Island, Korea, 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)*. Beijing, China, pages 1733–1743.

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

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, Kevin Duh, Manaal Faruqui, Cynthia Gan, Dan Garrette, Yangfeng Ji, Lingpeng Kong, Adhiguna Kuncoro, Gaurav Kumar, Chaitanya Malaviya, Paul Michel, Yusuke Oda, Matthew Richardson, Naomi Saphra, Swabha Swayamdipta, and Pengcheng Yin. 2017. Dynet: The dynamic neural network toolkit. *arXiv preprint arXiv:1701.03980* .

Wenzhe Pei, Tao Ge, and Baobao Chang. 2014. Max-margin tensor neural network for Chinese word segmentation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Baltimore, Maryland, pages 293–303.

Fuchun Peng, Fangfang Feng, and Andrew McCallum. 2004. Chinese segmentation and new word detection using conditional random fields. In *Proceedings of the 20th international conference on Computational Linguistics*. Geneva, Switzerland, pages 562–568.

Yanjun Qi, Sujatha G Das, Ronan Collobert, and Jason Weston. 2014. Deep learning for character-based information extraction. In *Advances in Information Retrieval*, pages 668–674.

Weiwei Sun. 2010. Word-based and character-based word segmentation models: Comparison and combination. In *Proceedings of the 23rd International Conference on Computational Linguistics*. Beijing, China, pages 1211–1219.

Xu Sun, Houfeng Wang, and Wenjie Li. 2012. Fast online training with frequency-adaptive learning rates for Chinese word segmentation and new word detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*. Jeju Island, Korea, pages 253–262.

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. 2012. LSTM neural networks for language modeling. In *13th Annual Conference of the International Speech Communication Association*. Portland, Oregon, USA, pages 194–197.

Ben Taskar, Vassil Chatalbashev, Daphne Koller, and Carlos Guestrin. 2005. Learning structured prediction models: A large margin approach. In *Proceedings of the 22nd international conference on Machine learning*. Bonn, Germany, pages 896–903.

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*. Jeju Island, Korea, pages 168–171.

Mengqiu Wang, Rob Voigt, and Christopher D. Manning. 2014. Two knives cut better than one: Chinese word segmentation with dual decomposition. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Baltimore, Maryland, pages 193–198.

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)*. Berlin, Germany, pages 567–572.

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

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)*. Sofia, Bulgaria, pages 770–779.

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*. Seattle, USA, pages 311–321.

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*. Berlin, Germany, pages 421–431.

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*. Prague, Czech Republic, pages 840–847.

Yue Zhang and Stephen Clark. 2011. Syntactic processing using the generalized perceptron and beam search. *Computational linguistics* 37(1):105–151.

Hai Zhao, Deng Cai, Yang Xin, Wang Yuzhu, and Zhongye Jia. 2017. A hybrid model for Chinese spelling check. *ACM Transactions on Asian and Low-Resource Language Information Processing* 16(3).

Hai Zhao, Chang-Ning Huang, Mu Li, and Bao-Liang Lu. 2006. Effective tag set selection in Chinese word segmentation via conditional random field modeling. In *The 20th Pacific Asia Conference on Language, Information and Computation*. Wuhan, China, pages 87–94.

Hai Zhao, Chang-Ning Huang, Mu Li, and Bao-Liang Lu. 2010. A unified character-based tagging framework for Chinese word segmentation. *ACM Transactions on Asian Language Information Processing* 16(21).

Hai Zhao and Chunyu Kit. 2008a. An empirical comparison of goodness measures for unsupervised Chinese word segmentation with a unified framework. In *Proceedings of the Third International Joint Conference on Natural Language Processing*. Hyderabad, India, pages 9–16.

Hai Zhao and Chunyu Kit. 2008b. Exploiting unlabeled text with different unsupervised segmentation criteria for Chinese word segmentation. *Research in Computing Science* 33:93–104.

Hai Zhao and Chunyu Kit. 2008c. Unsupervised segmentation helps supervised learning of character tagging for word segmentation and named entity recognition. In *The Sixth SIGHAN Workshop on Chinese Language Processing*. Hyderabad, India, pages 106–111.

Hai Zhao and Chunyu Kit. 2011. Integrating unsupervised and supervised word segmentation: The role of goodness measures. *Information Sciences* 181(1):163–183.

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*. Seattle, Washington, USA, pages 647–657.