# Evaluating the Utility of Hand-crafted Features in Sequence Labelling*

**Minghao Wu**[♠♡†] **Fei Liu**[♠] **Trevor Cohn**[♠]
[♠]The University of Melbourne, Victoria, Australia
[♡]JD AI Research, Beijing, China
wuminghao@jd.com
fliu3@student.unimelb.edu.au
t.cohn@unimelb.edu.au



## Abstract

Conventional wisdom is that hand-crafted features are redundant for deep learning models, as they already learn adequate representations of text automatically from corpora. In this work, we test this claim by proposing a new method for exploiting handcrafted features as part of a novel hybrid learning approach, incorporating a feature auto-encoder loss component. We evaluate on the task of named entity recognition (NER), where we show that including manual features for part-of-speech, word shapes and gazetteers can improve the performance of a neural CRF model. We obtain a $F_1$ of 91.89 for the CoNLL-2003 English shared task, which significantly outperforms a collection of highly competitive baseline models. We also present an ablation study showing the importance of auto-encoding, over using features as either inputs or outputs alone, and moreover, show including the autoencoder components reduces training requirements to 60%, while retaining the same predictive accuracy.

## 1 Introduction

Deep neural networks have been proven to be a powerful framework for natural language processing, and have demonstrated strong performance on a number of challenging tasks, ranging from machine translation (Cho et al., 2014b,a), to text categorisation (Zhang et al., 2015; Joulin et al., 2017; Liu et al., 2018b). Not only do such deep models outperform traditional machine learning methods, they also come with the benefit of not requiring difficult feature engineering. For instance, both Lample et al. (2016) and Ma and Hovy (2016) propose end-to-end models for sequence labelling task and achieve state-of-the-art results.

Orthogonal to the advances in deep learning is the effort spent on feature engineering. A representative example is the task of named entity recognition (NER), one that requires both lexical and syntactic knowledge, where, until recently, most models heavily rely on statistical sequential labelling models taking in manually engineered features (Florian et al., 2003; Chieu and Ng, 2002; Ando and Zhang, 2005). Typical features include POS and chunk tags, prefixes and suffixes, and external gazetteers, all of which represent years of accumulated knowledge in the field of computational linguistics.

The work of Collobert et al. (2011) started the trend of feature engineering-free modelling by learning internal representations of compositional components of text (e.g., word embeddings). Subsequent work has shown impressive progress through capturing syntactic and semantic knowledge with dense real-valued vectors trained on large unannotated corpora (Mikolov et al., 2013a,b; Pennington et al., 2014). Enabled by the powerful representational capacity of such embeddings and neural networks, feature engineering has largely been replaced with taking off-the-shelf pre-trained word embeddings as input, thereby making models fully end-to-end and the research focus has shifted to neural network architecture engineering.

More recently, there has been increasing recognition of the utility of linguistic features (Li et al., 2017; Chen et al., 2017; Wu et al., 2017; Liu et al., 2018a) where such features are integrated to improve model performance. Inspired by this, taking NER as a case study, we investigate the utility of hand-crafted features in deep learning models, challenging conventional wisdom in an attempt to refute the utility of manually-engineered features. Of particular interest to this paper is the work by Ma and Hovy (2016) where they

*https://github.com/minghao-wu/CRF-AE
†Work carried out at The University of Melbourne

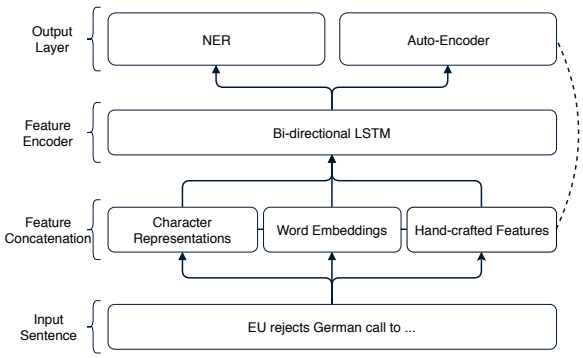


Figure 1: Main architecture of our neural network. Character representations are extracted by a character-level CNN. The dash line indicates we use an auto-encoder loss to reconstruct hand-crafted features.

introduce a strong end-to-end model combining a bi-directional Long Short-Term Memory (Bi-LSTM) network with Convolutional Neural Network (CNN) character encoding in a Conditional Random Field (CRF). Their model is highly capable of capturing not only word- but also character-level features. We extend this model by integrating an auto-encoder loss, allowing the model to take hand-crafted features as input and re-construct them as output, and show that, even with such a highly competitive model, incorporating linguistic features is still beneficial. Perhaps the closest to this study is the works by Ammar et al. (2014) and Zhang et al. (2017), who show how CRFs can be framed as auto-encoders in unsupervised or semi-supervised settings.

With our proposed model, we achieve strong performance on the CoNLL 2003 English NER shared task with an $F_1$ of 91.89, significantly outperforming an array of competitive baselines. We conduct an ablation study to better understand the impacts of each manually-crafted feature. Finally, we further provide an in-depth analysis of model performance when trained with varying amount of data and show that the proposed model is highly competent with only 60% of the training set.

## 2 Methodology

In this section, we first outline the model architecture, then the manually crafted features, and finally how they are incorporated into the model.

### 2.1 Model Architecture

We build on a highly competitive sequence labelling model, namely Bi-LSTM-CNN-CRF, first introduced by Ma and Hovy (2016). Given an input sequence of $\mathbf{x} = \{x_1, x_2, \ldots, x_T\}$ of length $T$, the model is capable of tagging each input with a predicted label $\hat{y}$, resulting in a sequence of $\hat{\mathbf{y}} = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_T\}$ closely matching the gold label sequence $\mathbf{y} = \{y_1, y_2, \ldots, y_T\}$. Here, we extend the model by incorporating an auto-encoder loss taking hand-crafted features as in/output, thereby forcing the model to preserve crucial information stored in such features and allowing us to evaluate the impacts of each feature on model performance. Specifically, our model, referred to as Neural-CRF+AE, consists of four major components: (1) a character-level CNN (char-CNN); (2) a word-level bi-directional LSTM (Bi-LSTM); (3) a conditional random field (CRF); and (4) an auto-encoder auxiliary loss. An illustration of the model architecture is presented in Figure 1.

**Char-CNN.** Previous studies (Santos and Zadrozny, 2014; Chiu and Nichols, 2016; Ma and Hovy, 2016) have demonstrated that CNNs are highly capable of capturing character-level features. Here, our character-level CNN is similar to that used in Ma and Hovy (2016) but differs in that we use a ReLU activation (Nair and Hinton, 2010).[1]

**Bi-LSTM.** We use a Bi-LSTM to learn contextual information of a sequence of words. As inputs to the Bi-LSTM, we first concatenate the pre-trained embedding of each word $\boldsymbol{w}_i$ with its character-level representation $\boldsymbol{c}_{w_i}$ (the output of the char-CNN) and a vector of manually crafted features $\boldsymbol{f}_i$ (described in Section 2.2):

$$\overrightarrow{\boldsymbol{h}}_i = \overrightarrow{\text{LSTM}}(\overrightarrow{\boldsymbol{h}}_{i-1}, [\boldsymbol{w}_i; \boldsymbol{c}_{w_i}; \boldsymbol{f}_i]) \quad (1)$$

$$\overleftarrow{\boldsymbol{h}}_i = \overleftarrow{\text{LSTM}}(\overleftarrow{\boldsymbol{h}}_{i+1}, [\boldsymbol{w}_i; \boldsymbol{c}_{w_i}; \boldsymbol{f}_i]), \quad (2)$$

where $[;]$ denotes concatenation. The outputs of the forward and backward pass of the Bi-LSTM is then concatenated $\boldsymbol{h}_i = [\overrightarrow{\boldsymbol{h}}_i; \overleftarrow{\boldsymbol{h}}_i]$ to form the output of the Bi-LSTM, where dropout is also applied.

**CRF.** For sequence labelling tasks, it is intuitive and beneficial to utilise information carried between neighbouring labels to predict the best sequence of labels for a given sentence. Therefore,

[1] While the hyperbolic tangent activation function results in comparable performance, the choice of ReLU is mainly due to faster convergence.

| x | *U.N.* | *official* | *Ekeus* | *heads* | *for* | *Baghdad* | . |
|---|---|---|---|---|---|---|---|
| POS | NNP | NN | NNP | VBZ | IN | NNP | . |
| Word shape | X.X. | xxxx | Xxxxx | xxxx | xxx | Xxxxx | . |
| Dependency tags | compound | compound | compound | ROOT | prep | pobj | punct |
| Gazetteer | O | O | PER | O | O | LOC | O |
| y | B-ORG | O | B-PER | O | O | B-LOC | O |

Table 1: Example sentence (top), showing the different types of linguistic features used in this work as additional inputs and auxiliary outputs (middle), and its labelling (bottom).

we employ a conditional random field layer (Lafferty et al., 2001) taking as input the output of the Bi-LSTM $\boldsymbol{h}_i$. Training is carried out by maximising the log probability of the gold sequence: $\mathcal{L}_{CRF} = \log p(\mathbf{y}|\mathbf{x})$ while decoding can be efficiently performed with the Viterbi algorithm.

**Auto-encoder loss.** Alongside sequence labelling as the primary task, we also deploy, as auxiliary tasks, three auto-encoders for reconstructing the hand-engineered feature vectors. To this end, we add multiple independent fully-connected dense layers, all taking as input the Bi-LSTM output $\boldsymbol{h}_i$ with each responsible for reconstructing a particular type of feature: $\hat{\boldsymbol{f}}_i^t = \sigma(\boldsymbol{W}^t \boldsymbol{h}_i)$ where $\sigma$ is the sigmoid activation function, $t$ denotes the type of feature, and $\boldsymbol{W}^t$ is a trainable parameter matrix. More formally, we define the auto-encoder loss as:

$$\mathcal{L}_{AE}^t = \sum_{i=0}^{T} \text{XEntropy}(\boldsymbol{f}_i^t, \hat{\boldsymbol{f}}_i^t)\,. \tag{3}$$

**Model training.** Training is carried out by optimising the joint loss:

$$\mathcal{L} = \mathcal{L}_{CRF} + \sum_t \lambda^t \mathcal{L}_{AE}^t\,, \tag{4}$$

where, in addition to $\mathcal{L}_{CRF}$, we also add the auto-encoder loss, weighted by $\lambda^t$. In all our experiments, we set $\lambda^t$ to 1 for all $t$s.

### 2.2 Hand-crafted Features

We consider three categories of widely used features: (1) POS tags; (2) word shape; and (3) gazetteers and present an example in Table 1. While POS tags carry syntactic information regarding sentence structure, the word shape feature focuses on a more fine-grained level, encoding character-level knowledge to complement the loss of information caused by embedding lookup, such as capitalisation. Both features are based on the implementation of spaCy.[2] For the gazetteer feature, we focus on *PERSON* and *LOCATION* and compile a list for each. The *PERSON* gazetteer is collected from U.S. census 2000, U.S. census 2010 and DBpedia whereas GeoNames is the main source for *LOCATION*, taking in both official and alternative names. All the tokens on both lists are then filtered to exclude frequently occurring common words.[3] Each category is converted into a one-hot sparse feature vector $\boldsymbol{f}_i^t$ and then concatenated to form a multi-hot vector $\boldsymbol{f}_i = [\boldsymbol{f}_i^{\text{POS}}; \boldsymbol{f}_i^{\text{shape}}; \boldsymbol{f}_i^{\text{gazetteer}}]$ for the $i$-th word. In addition, we also experimented with including the label of the incoming dependency edge to each word as a feature, but observed performance deterioration on the development set. While we still study and analyse the impacts of this feature in Table 3 and Section 3.2, it is excluded from our model configuration (not considered as part of $\boldsymbol{f}_i$ unless indicated otherwise).

## 3 Experiments

In this section, we present our experimental setup and results for name entity recognition over the CoNLL 2003 English NER shared task dataset (Tjong Kim Sang and De Meulder, 2003).

### 3.1 Experimental Setup

**Dataset.** We use the CoNLL 2003 NER shared task dataset, consisting of 14,041/3,250/3,453 sentences in the training/development/test set respectively, all extracted from Reuters news articles during the period from 1996 to 1997. The dataset is annotated with four categories of name entities: *PERSON, LOCATION, ORGANIZATION* and *MISC*. We use the `IOBES` tagging scheme, as previous study have shown that this scheme provides a modest improvement to the model performance (Ratinov and Roth, 2009; Chiu and Nichols, 2016; Lample et al., 2016; Ma and Hovy, 2016).

[2] https://spacy.io/

[3] Gazetteer data is included in the code release.

**Model configuration.** Following the work of Ma and Hovy (2016), we initialise word embeddings with GloVe (Pennington et al., 2014) (300-dimensional, trained on a 6B-token corpus). Character embeddings are 30-dimensional and randomly initialised with a uniform distribution in the range $[-\sqrt{\frac{3}{dim}}, +\sqrt{\frac{3}{dim}}]$. Parameters are optimised with stochastic gradient descent (SGD) with an initial learning rate of $\eta = 0.015$ and momentum of $0.9$. Exponential learning rate decay is applied every 5 epochs with a factor of $0.8$. To reduce the impact of exploding gradients, we employ gradient clipping at $5.0$ (Pascanu et al., 2013).

We train our models on a single GeForce GTX TITAN X GPU. With the above hyper-parameter setting, training takes approximately $8$ hours for a full run of $40$ epochs.

**Evaluation.** We measure model performance with the official CoNLL evaluation script and report span-level named entity F-score on the test set using early stopping based on the performance on the validation set. We report average F-scores and standard deviation over 5 runs for our model.

**Baseline.** In addition to reporting a number of prior results of competitive baseline models, as listed in Table 2, we also re-implement the Bi-LSTM-CNN-CRF model by Ma and Hovy (2016) (referred to as Neural-CRF in Table 2) and report its average performance.

### 3.2 Results

The experimental results are presented in Table 2. Observe that Neural-CRF+AE, trained either on the training set only or with the addition of the development set, achieves substantial improvements in F-score in both settings, superior to all but one of the benchmark models, highlighting the utility of hand-crafted features incorporated with the proposed auto-encoder loss. Compared against the Neural-CRF, a very strong model in itself, our model significantly improves performance, showing the positive impact of our technique for exploiting manually-engineered features. Although Peters et al. (2018) report a higher F-score using their ELMo embedding technique, our approach here is orthogonal, and accordingly we would expect a performance increase if we were to incorporate their ELMo representations into our model.

**Ablation Study** To gain a better understanding of the impacts of each feature, we perform an ablation study and present the results in Table 3. We observe performance degradation when eliminating POS, word shape and gazetteer features, showing that each feature contributes to NER performance beyond what is learned through deep learning alone. Interestingly, the contribution of gazetteers is much less than that of the other features, which is likely due to the noise introduced in the matching process, with many incorrectly identified false positives.

Including features based on dependency tags into our model decreases the performance slightly. This might be a result of our simple implementation (as illustrated in Table 1), which does not include dependency direction, nor parent-child relationships.

Next, we investigate the impact of different means of incorporating manually-engineered features into the model. To this end, we experiment with three configurations with features as: (1) input only; (2) output only (equivalent to multi-task learning); and (3) both input and output (Neural-CRF+AE) and present the results in Table 4. Simply using features as either input or output only improves model performance slightly, but insignificantly so. It is only when features are incorporated with the proposed auto-encoder loss do we observe a significant performance boost.

| Model | $F_1$ |
|---|---|
| Chieu and Ng (2002) | 88.31 |
| Florian et al. (2003) | 88.76 |
| Ando and Zhang (2005) | 89.31 |
| Collobert et al. (2011) | 89.59 |
| Huang et al. (2015) | 90.10 |
| Passos et al. (2014) | 90.90 |
| Lample et al. (2016) | 90.94 |
| Luo et al. (2015) | 91.20 |
| Ma and Hovy (2016) | 91.21 |
| Yang et al. (2017) | 91.62 |
| Peters et al. (2018) | 90.15 |
| Peters et al. (2018)+ELMo | **92.22** ($\pm$ 0.10) |
| Neural-CRF‡ | 91.06 ($\pm$ 0.18) |
| Neural-CRF+AE‡$*$ | 91.89 ($\pm$ 0.23) |
| Ratinov and Roth (2009)† | 90.80 |
| Chiu and Nichols (2016)† | 91.62 |
| Neural-CRF+AE† ‡ | **92.29** ($\pm$ 0.20) |

Table 2: NER Performance on the CoNLL 2003 English NER shared task test set. **Bold** highlights best performance. † marks models trained on both the training and development sets. ‡ indicates average performance over 5 runs. $*$ indicates statistical significance on the test set against Neural-CRF by two-sample Student's t-test at level $\alpha = 0.05$.

| Model | Dev $F_1$ | Test $F_1$ |
|---|---|---|
| Neural-CRF+AE | **94.87** ($\pm$ 0.21) | **91.89** ($\pm$ 0.23) |
| $-$ POS tagging$*$ | 94.78 ($\pm$ 0.17) | 91.30 ($\pm$ 0.28) |
| $-$ word shape$*$ | 94.83 ($\pm$ 0.31) | 91.36 ($\pm$ 0.30) |
| $-$ gazetteer | 94.85 ($\pm$ 0.20) | 91.80 ($\pm$ 0.19) |
| $+$ dependencies | 94.74 ($\pm$ 0.16) | 91.66 ($\pm$ 0.18) |

Table 3: Ablation study. Average performance over 5 runs with standard deviation. $+$ and $-$ denote adding and removing a particular feature (to/from Neural-CRF+AE trained on the training set only with POS tagging, word shape and gazetteer features). $*$ indicates statistical significance on the test set against Neural-CRF+AE by two-sample Student's t-test at level $\alpha = 0.05$. Note that in this table, $*$ measures the drop in performance.

| Model | Dev $F_1$ | Test $F_1$ |
|---|---|---|
| Neural-CRF | 94.53 ($\pm$ 0.21) | 91.06 ($\pm$ 0.18) |
| $+$ input | 94.63 ($\pm$ 0.23) | 91.17 ($\pm$ 0.25) |
| $+$ output | 94.69 ($\pm$ 0.22) | 91.23 ($\pm$ 0.19) |
| $+$ input & output$*$ | **94.87** ($\pm$ 0.21) | **91.89** ($\pm$ 0.23) |

Table 4: Average performance of Neural-CRF with different features configurations over 5 runs with standard deviation. Note that $+$ input & output = Neural-CRF+AE. $*$ indicates statistical significance on the test set against Neural-CRF by two-sample Student's t-test at level $\alpha = 0.05$.

**Training Requirements** Neural systems typically require a large amount of annotated data. Here we measure the impact of training with varying amount of annotated data, as shown in Figure 2. Wtih the proposed model architecture, the amount of labelled training data can be drastically reduced: our model, achieves comparable performance against the baseline Neural-CRF, with as little as 60% of the training data. Moreover, as we increase the amount of training text, the performance of Neural-CRF+AE continues to improve.

**Hyperparameters** Three extra hyperparameters are introduced into our model, controlling the weight of the autoencoder loss relative to the CRF loss, for each feature type. Figure 3 shows the effect of each hyperparameter on test performance. Observe that setting $\lambda_i = 1$ gives strong performance, and that the impact of the gazetteer is less marked than the other two feature types. While increasing $\lambda$ is mostly beneficial, performance drops if the $\lambda s$ are overly large, that is, the auto-encoder loss overwhelms the main prediction task.

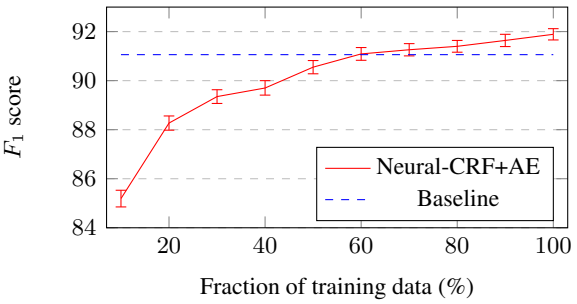


Figure 2: Comparing the Neural-CRF+AE (red solid line) trained with varying amounts of data vs. a Neural-CRF baseline (blue dashed line), trained on the full training set. Performance averaged over 5 runs, and error bars show $\pm$ 1 std.dev.

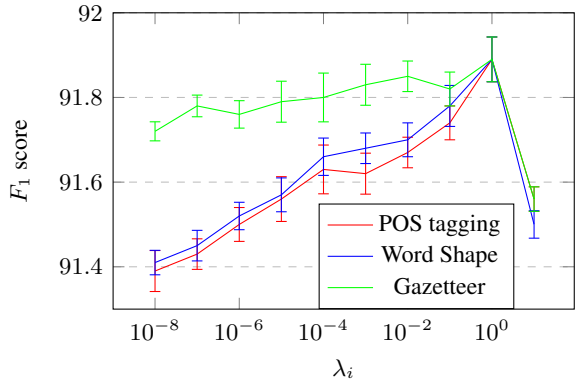


Figure 3: Effect of hyperparameter values on model performance. Each curve shows the effect of $\lambda_i$, for feature type $i$, with all other $\lambda_j = 1$, $j \neq i$. Performance averaged over 5 runs, and error bars show $\pm$ 1 variance.

## 4 Conclusion

In this paper, we set out to investigate the utility of hand-crafted features. To this end, we have presented a hybrid neural architecture to validate this hypothesis extending a Bi-LSTM-CNN-CRF by incorporating an auto-encoder loss to take manual features as input and then reconstruct them. On the task of named entity recognition, we show significant improvements over a collection of competitive baselines, verifying the value of such features. Lastly, the method presented in this work can also be easily applied to other tasks and models, where hand-engineered features provide key insights about the data.

## References

Waleed Ammar, Chris Dyer, and Noah A Smith. 2014. Conditional random field autoencoders for unsupervised structured prediction. In *Proceedings of the 27th International Conference on Neural Informa-*

*tion Processing Systems (NIPS 2014)*, pages 3311–3319.

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6(Nov):1817–1853.

Huadong Chen, Shujian Huang, David Chiang, and Jiajun Chen. 2017. Improved neural machine translation with a syntax-aware encoder and decoder. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 1936–1945.

Hai Leong Chieu and Hwee Tou Ng. 2002. Named entity recognition: A maximum entropy approach using global information. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002)*, pages 1–7.

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014a. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation (SSST-8 2014)*, pages 103–111.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014b. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 1724–1734.

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

Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. 2003. Named entity recognition through classifier combination. In *Proceedings of the Seventh Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2003)*, pages 168–171.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017)*, pages 427–431.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning (ICML 2001)*, pages 282–289.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2016)*, pages 260–270.

Junhui Li, Deyi Xiong, Zhaopeng Tu, Muhua Zhu, Min Zhang, and Guodong Zhou. 2017. Modeling source syntax for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 688–697.

Fei Liu, Trevor Cohn, and Timothy Baldwin. 2018a. Narrative modeling with memory chains and semantic supervision. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL 2018)*, pages 278–284.

Fei Liu, Trevor Cohn, and Timothy Baldwin. 2018b. Recurrent entity networks with delayed memory update for targeted aspect-based sentiment analysis. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2018)*, pages 278–283.

Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaiqing Nie. 2015. Joint entity recognition and disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 879–888.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 1064–1074.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *Proceedings of the 1st International Conference on Learning Representations (ICLR 2013)*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS 2013)*, pages 3111–3119.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML 2010)*, pages 807–814.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning (ICML 2013)*, pages 1310–1318.

Alexandre Passos, Vineet Kumar, and Andrew McCallum. 2014. Lexicon infused phrase embeddings for named entity resolution. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning (CoNLL 2014)*, pages 78–86.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2018)*, pages 2227–2237.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009)*, pages 147–155.

Cicero D Santos and Bianca Zadrozny. 2014. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31st International Conference on Machine Learning (ICML 2014)*, pages 1818–1826.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2003)*, pages 142–147.

Shuangzhi Wu, Dongdong Zhang, Nan Yang, Mu Li, and Ming Zhou. 2017. Sequence-to-dependency neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 698–707.

Jie Yang, Yue Zhang, and Fei Dong. 2017. Neural reranking for named entity recognition. In *Proceedings of the International Conference Recent Advances in Natural Language Processing (RANLP 2017)*, pages 784–792.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems (NIPS 2015)*, pages 649–657.

Xiao Zhang, Yong Jiang, Hao Peng, Kewei Tu, and Dan Goldwasser. 2017. Semi-supervised structured prediction with neural crf autoencoder. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 1701–1711.