Similarity Based Auxiliary Classifier for Named Entity Recognition

Shiyuan Xiao[1,2], **Yuanxin Ouyang**[1,2], **Wenge Rong**[1,2], **Jianxin Yang**[1,2], **Zhang Xiong**[1,2]
[1]Engineering Research Center of Advanced Computer Application Technology, Ministry of Education, Beihang University, Beijing, China
[2]School of Computer Science and Engineering, Beihang University, Beijing, China
{xiaoshiyuan, oyyx, w.rong, yjx17, xiongz}@buaa.edu.cn

Abstract

The segmentation problem is one of the fundamental challenges associated with name entity recognition (NER) tasks that aim to reduce the boundary error when detecting a sequence of entity words. A considerable number of advanced approaches have been proposed and most of them exhibit performance deterioration when entities become longer. Inspired by previous work in which a multi-task strategy is used to solve segmentation problems, we design a similarity based auxiliary classifier (SAC), which can distinguish entity words from non-entity words. Unlike conventional classifiers, SAC uses vectors to indicate tags. Therefore, SAC can calculate the similarities between words and tags, and then compute a weighted sum of the tag vectors, which can be considered a useful feature for NER tasks. Empirical results are used to verify the rationality of the SAC structure and demonstrate the SAC model's potential in performance improvement against our baseline approaches.

1 Introduction

Named entity recognition (NER) focuses on extracting a specific sequence and then classifying the sequence as a predefined category. As a fundamental and important task in natural language understanding and information extraction, NER can be considered a sequence labelling task, which includes part-of-speech (POS) tagging, chunking, and semantic role labelling (SRL) (Collobert et al., 2011). Owing to the growing popularity of deep learning techniques, a considerable number of neural network-based approaches and traditional conditional random fields (CRF) have been proposed and are widely used in many sequence labelling tasks to obtain magnificent results (Ye and Ling, 2018; Wu et al., 2018; Ghaddar and Langlais, 2018).

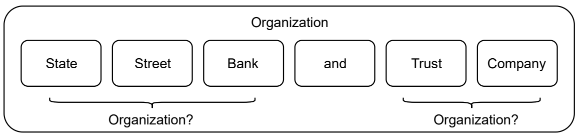


Figure 1: The segmentation problem in NER. "State Street Bank and Trust Company" is an organization entity with six words, it would be unsatisfactory if we consider "State Street Bank" and "Trust Company" to be two entities

To enhance NER's performance, significant effort has been devoted in this area, and one direction is to employ useful information at the input end. This mechanism can be realized by including hand-crafted features (Collobert et al., 2011; Passos et al., 2014; Huang et al., 2015). To eventually reduce heavy feature engineering, some researchers have argued for the exclusion of task-specific features and for the automatic mining of character level features and word level features (Ma and Hovy, 2016; Liu et al., 2018). Such end-to-end models can potentially improve performance with the exclusion of hand-crafted features.

While character and word level features demonstrate promising results, Ye and Ling (2018) determined that such methods (i.e. LM-BiLSTM-CRF (Liu et al., 2018)) will be relatively less effective for longer entity recognition. As indicated by Zhuo et al. (2016), NER is a segment-level task, and such NER models should be able to extract segment information while word-level labels might tend to learn more word level information than segment level information. Such limitations can manifest as entity boundary errors, referred to as the segmentation problem. This problem is illustrated in Figure 1.

There have been many attempts to address the segmentation problem. Some researchers employed segment level information, e.g., Semi-CRF

(SCRF) (Zhuo et al., 2016), to address this challenge. Though SCRF exhibits promising results in the case of longer entity recognition, it requires additional feature extractors and exhibits some limitations in the case of shorter entity recognition. Ye and Ling (2018) proposed using CRF jointly with SCRF to address this problem. On the other hand, researchers have also adopted a multi-task strategy to predict boundaries. For example, Stratos (2016) divides NER into a two-step task in which the boundary is first predicted and then the entity type of the boundary is determined. Though this method is easy to implement, it relies heavily on the accuracy of the first step. Another alternative thought is to employ auxiliary tasks to address this problem. For example, Aguilar et al. (2017) created two auxiliary tasks to help determine entity boundaries and types. However, the authors considered the auxiliary tasks to be supervisors, which means that the outputs of the auxiliary tasks are not fully utilized.

In this study, inspired by the research indicated previously, we design an auxiliary classification task that aims to distinguish entity words from non-entity words before executing the main NER task. Unlike prior studies, the outputs of the auxiliary task do not directly determine the entity boundaries but rather serve as features for assisting the main task. Therefore, the auxiliary task is more like an "advisor" rather than supervisor to the entire model. Our auxiliary classifier is different from conventional classifiers, which compute the probability of an entity word using the $softmax$ function. Instead, we initialize two trainable vectors for the classifier to identify "True" and "False" tags, and then we determine the similarity scores between each word and the two tag vectors to indicate the probability whether the word is an entity word. The hypothesis is that tag vectors can serve as helpful features that are shared by a majority of entity words and non-entity words in NER tasks. Finally, we obtain the weighted sum of the tag vectors by using the similarity scores. In this study, this process is referred to as a similarity based auxiliary classifier (SAC).[1] The empirical results verify the rationality of SAC's structure and demonstrate that the SAC model can reduce segmentation problems and exhibit good performance in recognizing entities of different lengths.

The contributions of this paper are as follows: 1) We introduce a similarity based auxiliary classifier (SAC), which allows the model to determine whether a word is an entity word before predicting its entity label to overcome the segmentation problem. 2) By analyzing the experimental results, we verify the rationality of SAC's structure and demonstrate the advantages of using the weighted sum of tag vectors compared to a list of scores denoting classification results. 3) Without using any hand-crafted features, NeuralCRF+SAC outperforms existing state-of-the-art end-to-end models on CoNLL 2003 and Ontonotes 5.0. When using external resource like ELMo, NeuralCRF+SAC still exhibits comparable results.

2 Related work

NER is a fundamental task, and traditional methods usually adopt statistical machine learning models e.g., Hidden Markov Models (HMM), Maximum-entropy Markov Models (MEMM), and Conditional Random Fields (CRF) (Bikei et al., 1998; McCallum et al., 2000; Lafferty et al., 2001). To improve overall performance, many hand-crafted features such as POS chunking tags and gazetteers are required and integrated into the task (Chieu and Ng, 2002; Florian et al., 2003; Ando and Zhang, 2005).

With the growing popularity of deep learning techniques, neural networks have been widely adopted for NER tasks. Collobert et al. (2011) initiated the trend of using neural networks for sequence labelling tasks. Huang et al. (2015) combined BiLSTM with CRF for NER tasks. Because of the powerful feature mining ability of neural networks, some end-to-end models employ neural structures to extract character features and obtain state-of-the-art results without using hand-crafted features (Lample et al., 2016; Ma and Hovy, 2016; Liu et al., 2018). Hand-crafted features such as POS tags, word capitalization, and lexicons are commonly used (Chiu and Nichols, 2016; Ghaddar and Langlais, 2018; Wu et al., 2018). It was recently proved that using contextualized representations from pretrained language models can significantly improve performance (Peters et al., 2017, 2018; Devlin et al., 2018).

To address the segmentation problem, several methods have been proposed, among which Semi-CRF(SCRF) is a successful solution for segment-

[1] code available at https://github.com/XiaoShiyuan/NCRF-SAC

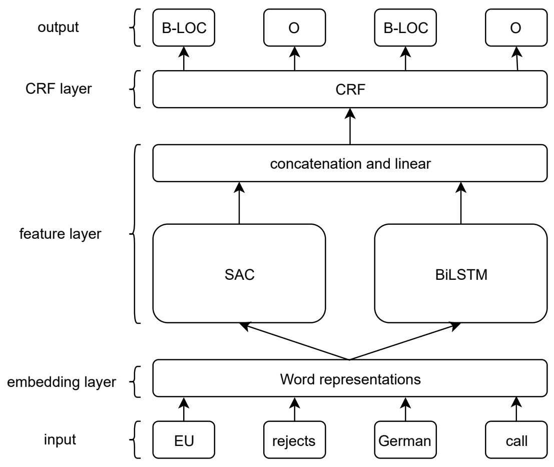


Figure 2: Overall architecture of NeuralCRF+SAC

level modeling (Zhuo et al., 2016). Ye and Ling (2018) proposed an improved version of SCRF, namely HSCRF. Using a multi-task strategy to combine HSCRF and CRF, the HSCRF model exhibits state-of-the-art results without using external resources. In particular, HSCRF resolves the segmentation problem by matching entity lengths. In addition, there are other approaches that consider multi-task learning to predict entity boundaries. Stratos (2016) divides NER into two tasks. First, CRF is used to predict the location and boundary of entities and to classify the entities. Aguilar et al. (2017) used the outputs of their feature extractor to perform segmentation, categorization, and then sequential classification.

Our method exploits the features of multi-task learning and proposes a similarity based auxiliary classifier (SAC). Compared to existing studies that usually consider auxiliary tasks to be supervisors, we also use the output of SAC to support the NER task. Meanwhile, we employ vectors to denote "True" and "False" tags, which contain more useful information compared to simply using "1" and "0" to denote tags.

3 Methodology

3.1 Overall Architecture

We implement the framework based on BiLSTM-CNN-CRF(NeuralCRF) (Ma and Hovy, 2016), a robust and widely used neural network in English NER tasks, and we build our SAC parallel to BiLSTM. The overall architecture of NeuralCRF+SAC is presented in Figure 2.

Embedding layer. Denoting input sequences as $X = \{x_1, x_2, ..., x_T\}$, we first obtain the word-level vector $W = \{w_1, w_2, ..., w_T\}$. Next, a Char-CNN is employed to obtain character-level features $C = \{c_{w_1}, c_{w_2}, ..., c_{w_T}\}$ (Santos and Zadrozny, 2014; Chiu and Nichols, 2016; Ma and Hovy, 2016). Finally, we obtain $[w_i; c_{w_i}]$ to denote x_i, where [;] indicates concatenation.

Feature layer. In this layer, we employ BiLSTM to extract contextual features and SAC to generate weighted vectors. The calculations can be expressed as follows:

$$\overrightarrow{h_i} = LSTM(\overrightarrow{h_{i-1}}, [w_i; c_{w_i}]) \tag{1}$$

$$\overleftarrow{h_i} = LSTM(\overleftarrow{h_{i-1}}, [w_i; c_{w_i}]) \tag{2}$$

$$h_{a_i} = SAC([w_i; c_{w_i}]) \tag{3}$$

Here, we use SAC to denote all operations of the classifier and we will introduce it in detail in Section 3.2. The three outputs are then concatenated and operated as $z_i = Linear([\overrightarrow{h_i}; \overleftarrow{h_i}; h_{a_i}])$, where $Linear$ denotes a linear calculation.

CRF layer. Because of the ability to utilize information from neighboring labels, CRF is widely employed in sequence tagging tasks. Denoting label sequences as $Y = \{y_1, y_2, ..., y_T\}$, the collection of all possible labels as $\mathcal{Y}$, and semantic feature sequence as $Z = \{z_1, z_2, ..., z_T\}$, CRF can be described as a family of conditional probabilities $p(y|Z)$:

$$p(y|Z) = \frac{\prod_{i=1}^{T} \phi_i(y_{i-1}, y_i, z_i)}{\sum_{y' \in \mathcal{Y}} \prod_{i=1}^{T} \phi_i(y'_{i-1}, y'_i, z_i)} \tag{4}$$

where $\phi_i(y_{i-1}, y_i, z_i) = exp(W_{y_{i-1},y} z_i + b_{y_{i-1},y_i})$ is the potential function and $Z = \{z_1, z_2, ..., z_T\}$ are the outputs of the feature layer.

For the training process, we minimize the negative log likelihood:

$$\mathcal{L}_{\mathcal{CRF}} = -\sum_i log(p(y_i|Z)) \tag{5}$$

while the Viterbi algorithm (Forney, 1973) is used to decode the best label sequence.

3.2 Similarity based Auxiliary Classifier

We believe that SAC can concentrate on one word and on the nearest neighbors of the word rather than on long-term dependency because we believe that information from the word itself and its nearest neighbors contribute the most in determining whether the word is an entity word. Compared to RNN, CNN exhibits a distinct advantage that

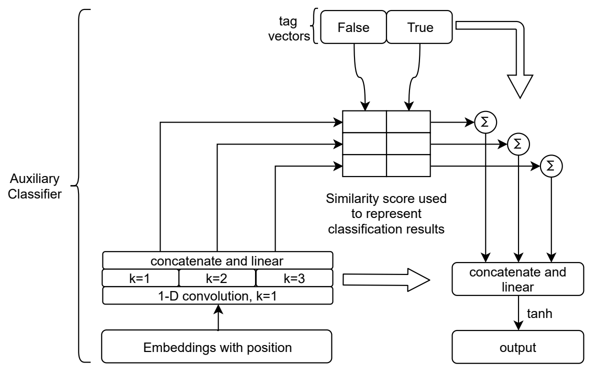


Figure 3: The architecture of SAC. Notice that the convolutional block (lower left) includes 4 1-dimentional convolutional neural networks, which share the same filter settings. Kernel size is marked out.

it can accurately control the length of the context, which satisfies our requirement. Therefore, SAC is based on CNN. The structure of this module is presented in Figure 3. To prevent the loss of positional information while using CNN, we implement a position encoder (Vaswani et al., 2017) to generate position vectors $p_i \in \mathbb{R}^d$, where d is configurable and p_i is concatenated with $[w_i; c_{w_i}]$ to form new embeddings $[w_i; c_{w_i}; p_i]$. The new embeddings are then sent to the convolutional block to extract uni-gram, bi-gram and tri-gram information. We denote the output of this convolutional block as $H_c = \{h_{c_1}, h_{c_2}, ..., h_{c_T}\}$, where $h_{c_i} \in \mathbb{R}^n$.

Intuitively, we adopt a multiplicative attention mechanism (Luong et al., 2015) for computing similarity. For the purpose of classifying entity words and non-entity words, we use $t_i, (i \in \{0, 1\})$ to denote non-entity words and entity words, respectively. Then, we randomly initialize two tag vectors $v_i \in \mathbb{R}^{d_h}, (i \in \{0, 1\})$ as semantic representations of $t_i, (i \in \{0, 1\})$. We express the computation as follows:

$$e_{ij} = v_i^T W_n h_{c_j} \tag{6}$$

$$a_{ij} = \frac{exp(e_{ij})}{\sum_{k=0}^{1} exp(e_{kj})} \tag{7}$$

$$s_j = \sum_{i=0}^{1} a_{ij} v_i \tag{8}$$

$$h_{a_j} = tanh(W_a[h_{c_j}; s_j] + b_a) \tag{9}$$

where $W_n \in \mathbb{R}^{d_h \times n}$, $W_a \in \mathbb{R}^{n \times (n+d_h)}$ and $b_a \in \mathbb{R}^n$. Innovatively, we use a_{ij}, the similarity score between i^{th} tag and h_{c_j} as the classification result.

Dataset		train	dev	test
CoNLL 2003	#tok	204567	51578	46666
	#ent	23499	5942	5648
Ontonotes 5.0	#tok	1088503	147724	152728
	#ent	81828	11066	11257
WNUT 2017	#tok	62729	15733	23394
	#ent	3160	1250	1589

Table 1: Statistics of these three datasets, #tok denotes tokens and #ent denotes entities.

For instance, if the j^{th} word is an entity word, its feature representation h_{c_j} must have a higher similarity score with v_1. Therefore, SAC operates in a manner that is similar to a "supervised cluster," separating entity words from non-entity words, and denoting two "clustering centers" with two tag vectors. According to Equation 8 and 9, classification information related to the j^{th} word can be expressed as a weighted sum s_j and is then concatenated with h_{c_j}. Finally, we obtain the output $H_a = \{h_{a_1}, h_{a_2}, ..., h_{a_T}\}$.

To train this auxiliary classifier, we define attention loss as follows:

$$\mathcal{L_A} = -\frac{1}{T}\sum_{j=1}^{T}\sum_{i=0}^{1} t_i log(a_{ij}) \tag{10}$$

By combining $\mathcal{L_{CRF}}$ and $\mathcal{L_A}$, we obtain the loss function:

$$\mathcal{L} = \mathcal{L_{CRF}} + \lambda \mathcal{L_A} \tag{11}$$

We obtain the best and most stable $\lambda = 0.05$ when using the IOBES tag scheme.

4 Experiments

4.1 Experiments Setup

Dataset. We performed experiments on CoNLL 2003 (Tjong Kim Sang and De Meulder, 2003), Ontonotes 5.0 (Hovy et al., 2006) and WNUT 2017 shared tasks (Derczynski et al., 2017). CoNLL 2003 is a widely used dataset comprising data extracted from Reuters news articles. Ontonotes 5.0 is a large-scale dataset comprising data collected from news, weblogs, etc. WNUT 2017 is a small dataset consisting primarily of emerging entities. All datasets have been separated into training/dev/test sets and they include 4/18/6 entity types, respectively. Table 1 presents some statistics of the 3 datasets.

Configuration. All models were implemented with TensorFlow [2]. We used GloVe (Pennington

[2]www.tensorflow.org

Module	Parameter	Value
Word embedding	dims	300
Char-CNN	char dims	100
	kernel size	3
	filters	50
BiLSTM	dims	300
Auxiliary Classifier	pos dims	40
	tag-vector dims	25
	CNN-block filters	300
Others	learning rate	0.001
	batch size	20
	epoch	50
	drop rate	0.5

Table 2: Hyper-parameters

structure	Accuracy	F1(max)
BiLSTM	96.97	91.20
CNN	**97.87**	**91.65**

Table 3: Comparing different structure of SAC. Accuracy indicates that of an auxiliary classification task and F1 for an NER task

et al., 2014) (300-dim, trained on 840B corpus[3]) pretrained word embeddings, and the same hyper-parameters on BiLSTM-CNN-CRF(NeuralCRF) and NeuralCRF+SAC. To determine performance using external resources, we included ELMo representations. All hyper-parameters, most of which are empirical parameters, are listed in Table 2 and an early stopping strategy was adopted. The main experiments were tested for 5 times to obtain the mean and standard deviation of F1 score.

Evaluation. After training, the latest 50 checkpoints were preserved and we obtained the checkpoint with the highest word-level F1-score to make predictions on the test set. Following this, we used the official CoNLL evaluation script on the CoNLL 2003 dataset and ontonotes 5.0 and the WNUT evaluation script on the WNUT 2017 dataset.

4.2 Development experiments

To verify the rationality of the structure of SAC and the role of the tag vectors, we designed two development experiments on the CoNLL 2003 dataset. One experiment compared the CNN structure in SAC with the BiLSTM structure. The other one compared SAC with a modified TextCNN.

Comparing with an RNN structure We believe that if a word is an entity word, its nearest neighbors are very likely to be entity words as well. Although there are many one-word entities, these entities always share an obvious feature such as capital letters, which can be easily captured by a char-CNN. On the contrary, if a word is a non-entity word, under most circumstances, there would be at most one entity word among the adjacent words. However, when considering long-term dependency, a word can be affected by several former or latter words, which could weaken the influence of adjacent words and introduce unnecessary noise. Therefore, we chose CNN instead of RNN in SAC and we set the max kernel size to 3.

[3]http://nlp.stanford.edu/data/glove.840B.300d.zip

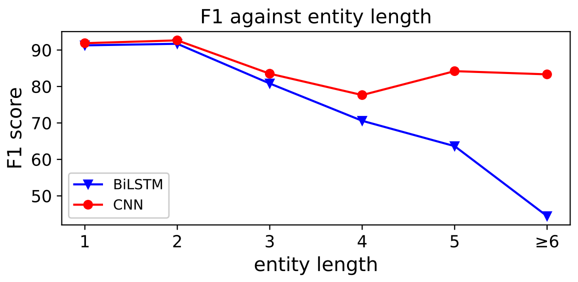


Figure 4: Comparing the ability of SAC with different structures to identify different length entities.

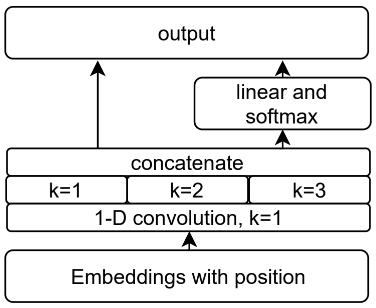


Figure 5: The structure of our TextCNN.

To verify if a CNN structure is more suitable in SAC, we replace the convolutional block with BiLSTM. In Table 3, we can observe that the maximum F1 value of the entire model is 91.65 when employing a CNN structure compared to 91.20 when using BiLSTM. To determine the reason for a large performance drop, we compare the performances of the two structures in the face of different entity lengths. From Figure 4, we can observe that if SAC uses a BiLSTM structure, the overall performance will still deteriorate as entities become long, indicating that SAC appears to be redundant for the entire model. Maybe it is because BiLSTM also exists in NeuralCRF, and using BiLSTM in SAC may cause NeuralCRF and SAC to obtain duplicate features.

structure	Accuracy	F1(max)
NeuralCRF+TextCNN	**98.29**	91.47
NeuralCRF+SAC	97.87	**91.65**

Table 4: Compare SAC with TextCNN. Accuracy is for the auxiliary classification task and F1 for NER task.

Models	type	F1(±std)
Zhuo et al. (2016)	reported	88.12
Lample et al. (2016)	reported	90.94
Ma and Hovy (2016)	reported	91.21
Strubell et al. (2017)	reported	90.54
Liu et al. (2018)	mean	91.24(±0.12)
	max	91.35
Ye and Ling (2018)	mean	91.38(±0.10)
	max	91.53
NeuralCRF	mean	91.22(±0.09)
	max	91.30
NeuralCRF+SAC	**mean**	**(91.48±0.07)**
	max	**91.65**

Table 5: Results on CoNLL 2003 NER task without external resources.

Comparing with TextCNN In addition, we employed a conventional classifier like TextCNN (Kim, 2014) to replace SAC. As is known, TextCNN predicts a set of scores for each sentence according to the number of categories. The position of the largest score indicates the category of the sentence. As is shown in Figure 5, to enable TextCNN to predict scores for each word, and to make a fair comparison with SAC, we remove the pooling layer of TextCNN and adopt the same convolutional structure as SAC. Meanwhile, both outputs of TextCNN and SAC include features mined by their convolutional layers. The difference lies in the other part of their outputs in which TextCNN includes a series of scores while SAC provides the weighted sum of the tag vectors.

Table 4 demonstrates that NeuralCRF+SAC obtains a maximum F1 score of 91.65 while NeuralCRF+TextCNN receives a score of 91.47. However, we can observe that TextCNN exhibits higher accuracy on its own classification task. We must declare that the accuracy of SAC has different meanings in terms of the accuracy of TextCNN. For TextCNN, an accuracy score of 98.29 only means that TextCNN assigns 98.29% of the words with the right tags. However, for SAC, an accuracy score of 97.87% also means that 97.87% of the words share common traits in which a majority of entity words are similar to the "True" tag vector, while the majority of non-entity words are similar to "False" tag vector. Therefore, because of the higher F1 score obtained from NeuralCRF+SAC, the two tag vectors can be viewed as features that are helpful for the NER task. Just like our baseline approach, NeuralCRF, uses char-CNN to mine character-level features. SAC automatically mines features that are shared by entity words or non-entity words.

Models	resources	F1(±std)
Chiu and Nichols (2016)	Lexicons	91.62(±0.33)
Peters et al. (2017)	TagLM†	91.93(±0.19)
Ghaddar and Langlais (2018)	gazetteer, LS	91.73
Wu et al. (2018)	POS,etc.	91.89(±0.23)
Peters et al. (2018)	ELMo†	92.22(±0.10)
Devlin et al. (2018)	BERT(BASE)†	92.40
Devlin et al. (2018)	**BERT(LARGE)**†	**92.80**
NeuralCRF	ELMo†	92.20(±0.06)
NeuralCRF+SAC	ELMo†	92.57(±0.07)

Table 6: Results of using the CoNLL 2003 NER task with ELMo representations. † denotes the language model trained on external resources.

Models	resources	F1(±std)
Shen et al. (2017)	None	86.63(±0.49)
Strubell et al. (2017)	None	86.84(±0.19)
NeuralCRF	None	86.65(±0.10)
NeuralCRF+SAC	**None**	**87.24(±0.07)**
Chiu and Nichols (2016)	Lexicons	86.28(±0.26)
Ghaddar and Langlais (2018)	gazetteer, LS	87.95
NeuralCRF	ELMo	88.41(±0.06)
NeuralCRF+SAC	**ELMo**	**89.05(±0.18)**

Table 7: Results on Ontonotes 5.0 dataset.

4.3 Results on CoNLL 2003 dataset

End-to-end. Table 5 presents the experimental results[4] on the CoNLL 2003 datasets without external resources. We can observe that our model exhibits a mean F1 score improvement of 0.26 compared to the base model and outperforms all existing end-to-end models.

With ELMo representations. In addition, be-

[4]The first author of (Liu et al., 2018) indicates errors pertaining to the original reported results because of some bugs and suggests citing fixed results. (https://github.com/LiyuanLucasLiu/LM-LSTM-CRF)

Models	F1(entity)	F1(surface)
Lin et al. (2017)	40.42	37.62
von Däniken et al. (2017)	40.78	39.33
Aguilar et al. (2017)	41.86	40.24
NeuralCRF	43.93	41.80
NeuralCRF+SAC	**44.77**	**43.29**

Table 8: Results of the WNUT 2017 shared task

Length	Ye and Ling (2018)	NeuralCRF+SAC	Δ
1	91.73	91.75	+0.02
2	92.03	92.70	+0.67
3	83.78	83.26	-0.52
4	77.27	78.57	+0.30
5	79.66	80.00	+0.34
$\geq$ 6	76.55	76.92	+0.37
overall	91.38	91.48	+0.10

Table 9: Comparing performance with Ye and Ling (2018).

cause of the popularity of contextualized representation, we introduce EMLo[5] in the embedding layer as a form of external resources and results, which are presented in Table 6. Observe that our model still exhibits a 0.37 F1 improvement over the base model, which verifies the effects of the SAC. Despite using BERT(LARGE), we still obtain state-of-the-art results, even though it requires a considerable number of resources to be expended. In contrast, our method is flexible and resources friendly.

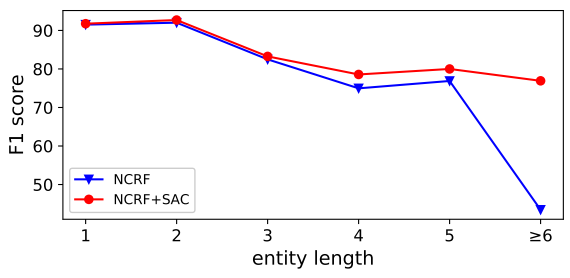
4.4 Results on Ontonotes 5.0 dataset

We present the experimental results on Ontonotes 5.0 in Table 7. Without using external resources, our model exhibits a mean F1 score of 87.24, which still exceeds the previous end-to-end model. When employing ELMo as external resources, we obtain a mean F1 score of 89.05. Comparing to NeuralCRF, SAC brings 0.59 and 0.64 F1 improvements respectively.

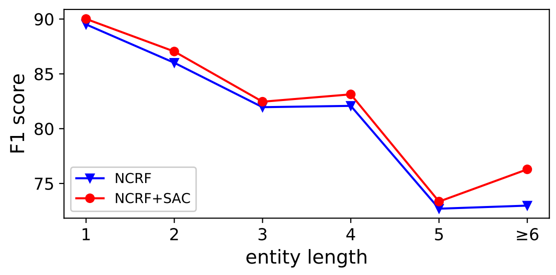
4.5 Results on WNUT 2017 shared task

Experimental results on the WNUT17 dataset are presented in Table 8. Owing to the facts that this task focuses on unusual and previously-unseen entities, all existing models incorporate some handcrafted features. We include ELMo representations in this experiment. SAC exhibits a 0.84 improvement in F1(entity) and 1.49 improvement in F1(surface).

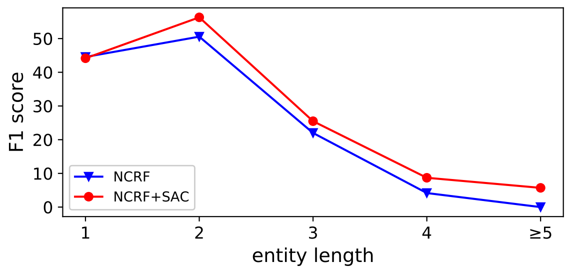
[5]https://allennlp.org/elmo



(a) F1 against entity length(CoNLL 2003)



(b) F1 against entity length(Ontonotes 5.0)



(c) F1 against entity length(WNUT 2017, with ELMo)

Figure 6: F1 against length test on all three datasets.

4.6 Analysis

Performance against entity length. We believe that before using CRF to predict the final label of a word, using a classifier to determine whether the word is an entity word helps in addressing problems related to segmentation, particularly when entities become long. Following Ye and Ling (2018)'s study, we compared the performance of NeuralCRF and NeuralCRF+SAC in identifying entities of different lengths on three datasets. The experimental results of the CoNLL 2003, Ontonotes 5.0, and WNUT 2017 datasets are presented in Figure 6. To eliminate the impact of external resources, we use the end-to-end approach as far as possible. But because the WNUT 2017 datasets are relatively small and most of them are emerging entities, which makes both

Ground truth	**Models**	**Results**
the Financial Times of Londom	NeuralCRF	correspondent for the Financial Times(ORG) of London(GPE)...
	NeuralCRF+SAC	correspondent for the Financial Times of Londom(ORG)...
What Else is Making News	NeuralCRF	Watch What Else(creative-work) is Making News
	NeuralCRF+SAC	Watch What Else is Making News(creative-work)
Monopolies and Mergers Commission	NeuralCRF	...Monopolies(ORG) and Mergers Commission(ORG) unless the carriers complied...
	NeuralCRF+SAC	...Monopolies and Mergers Commission(ORG) unless the carriers complied...

Table 10: Three examples, identified entities are highlighted

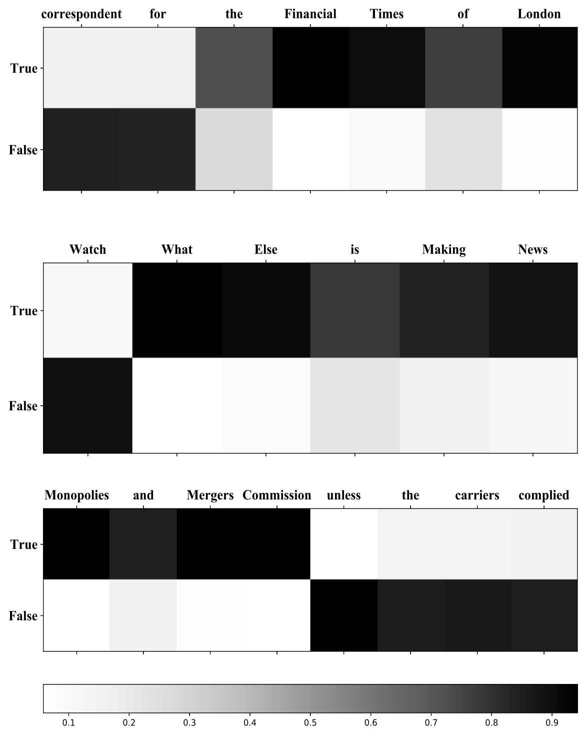


Figure 7: Similarity scores of three examples

models incapable of recognizing entities consisting of 4 or more words, we employed ELMo in WNUT 2017.

Observe that Figure 6(a) indicates a big improvement when entity lengths are equal to or greater than 4 and Figures 6(b) and 6(c) illustrate even improvements against different lengths.

In addition, we compared NeuralCRF+SAC with Ye and Ling (2018)' work in Table 9. The two models perform comparably when the entity length is less than 4, but NeuralCRF+SAC exhibits even improvements when the entity length is equal to or larger than 4, which demonstrates that SAC can improve the overall performance in long entity recognition.

Case study. We examined the manner in which SAC handles long entities. Table 10 presents three cases from 3 datasets, respectively. We observe that these 3 cases are quite similar and NeuralCRF makes almost the same mistake on each of them. Generally speaking, character level features are quite powerful for the NER task that words written in uppercase letters are easily recognized as entity words. In addition, if all the letters of a word are lowercase letters, they are more likely to be classified as non-entity words. Therefore, we can observe that cases in which meeting words like “of”, “is” and “and” are too normal to be considered entity words, it is very easy for NeuralCRF to predict an “O” label for these words. In fact, many long entities exhibit a pattern in which almost all words are written in uppercase letters except one. Therefore, it is common that the segmentation problem grows in severity as entities become longer. However, these three examples indicate that SAC can address this problem to some extent. We obtain the similarity scores of these examples and present them in Figure 7. It is obvious that SAC assigns a high similarity score to “of”, “is” and “and” even though they hardly include any features of entity words.

A possible reason is that the CNN structure of SAC plays a role. As discussed in Section 4.2, CNN can control the length of the context. When the kernel sizes are set to 1, 2 and 3, CNN layers can extract uni-gram, bi-gram and tri-gram features that enable SAC to learn a considerable number of samples that exhibit the same structure as “Time of London.” Therefore, SAC reduces truncation errors for long entities caused by lowercase words.

5 Conclusion

In this work, we extended the BiLSTM-CNN-CRF architecture with SAC to address the segmentation problem. By computing the similarity scores between words and two tag vectors, respectively, SAC provides the weighted sum of tag vectors together with features it mined to support the main NER task. Because of the higher accuracy of SAC, the two tag vectors can be considered features that are shared between entity words and non-entity words. By reducing the boundary errors, SAC addresses the segmentation problem and improves overall performance, particularly when entities become long. Experimental results demonstrate that NeuralCRF+SAC surpasses previous state-of-the-art end-to-end models on the CoNLL 2003 and Ontonotes 5.0 datasets without using external resources. Moreover, when including ELMo, SAC still improves overall performance and is comparable to current state-of-the-art models.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (No. 61977002)

References

Gustavo Aguilar, Suraj Maharjan, A Pastor López-Monroy, and Thamar Solorio. 2017. A multi-task approach for named entity recognition in social media data. *W-NUT 2017*, page 148.

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.

Daniel M. Bikei, Scott Miller, Richard Schwartz, and Ralph Weischedel. 1998. Nymble: High-performance learning name-finder. *Anlp*, pages 194–201.

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-Volume 1*, pages 1–7. Association for Computational Linguistics.

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

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.

Pius von Däniken, AG SpinningBytes, and Mark Cieliebak. 2017. Transfer learning and sentence level features for named entity recognition on tweets. *W-NUT 2017*, page 166.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the wnut2017 shared task on novel and emerging entity recognition. *W-NUT 2017*, page 140.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. 2003. Named entity recognition through classifier combination. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 168–171. Association for Computational Linguistics.

G David Forney. 1973. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278.

Abbas Ghaddar and Phillippe Langlais. 2018. Robust lexical features for improved neural network named-entity recognition. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1896–1907.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. Ontonotes: The 90\% solution. In *Proceedings of the human language technology conference of the NAACL, Companion Volume: Short Papers*.

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

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

John Lafferty, Andrew Mccallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proceedings of Icml*, 3(2):282–289.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT*, pages 260–270.

Bill Y Lin, Frank F Xu, Zhiyi Luo, and Kenny Q Zhu. 2017. Multi-channel bilstm-crf model for emerging named entity recognition in social media. *W-NUT 2017*, page 160.

L. Liu, J. Shang, F. Xu, X. Ren, H. Gui, J. Peng, and J. Han. 2018. Empower sequence labeling with task-aware neural language model. In *AAAI*.

Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

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 (Volume 1: Long Papers)*, volume 1, pages 1064–1074.

Andrew McCallum, Dayne Freitag, and Fernando CN Pereira. 2000. Maximum entropy markov models for information extraction and segmentation. In *Icml*, volume 17, pages 591–598.

Alexandre Passos, Vineet Kumar, and Andrew McCallum. 2014. Lexicon infused phrase embeddings for named entity resolution. *arXiv preprint arXiv:1404.5367*.

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)*, pages 1532–1543.

Matthew Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1756–1765.

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, Volume 1 (Long Papers)*, volume 1, pages 2227–2237.

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-14)*, pages 1818–1826.

Yanyao Shen, Hyokun Yun, Zachary C Lipton, Yakov Kronrod, and Animashree Anandkumar. 2017. Deep active learning for named entity recognition. *arXiv preprint arXiv:1707.05928*.

Karl Stratos. 2016. Entity identification as multitasking. *arXiv preprint arXiv:1612.02706*.

Emma Strubell, Patrick Verga, David Belanger, and Andrew McCallum. 2017. Fast and accurate entity recognition with iterated dilated convolutions. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2670–2680.

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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Minghao Wu, Fei Liu, and Trevor Cohn. 2018. Evaluating the utility of hand-crafted features in sequence labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2850–2856.

Zhixiu Ye and Zhen-Hua Ling. 2018. Hybrid semi-markov crf for neural sequence labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 235–240.

Jingwei Zhuo, Yong Cao, Jun Zhu, Bo Zhang, and Zaiqing Nie. 2016. Segment-level sequence modeling using gated recursive semi-markov conditional random fields. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1413–1423.