UNDERSTANDING AND IMPROVING SEQUENCE-LABELING NER WITH SELF-ATTENTIVE LSTMS

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper improves upon the line of research that formulates named entity recognition (NER) as a sequence-labeling problem. We use so-called black-box long short-term memory (LSTM) encoders to achieve state-of-the-art results while providing insightful understanding of what the auto-regressive model learns with a parallel self-attention mechanism. Specifically, we decouple the sequence-labeling problem of NER into entity chunking, e.g., $Barack_B$ $Obama_E$ was_O $elected_O$, and entity typing, e.g., $Barack_{PERSON}$ $Obama_{PERSON}$ was_{NONE} $elected_{NONE}$, and analyze how the model learns to, or has difficulties in, capturing text patterns for each of the subtasks. The insights we gain then lead us to explore a more sophisticated deep cross-Bi-LSTM encoder, which proves better at capturing global interactions given both empirical results and a theoretical justification.

1 INTRODUCTION

Named entity recognition is an important task in information extraction in which we seek to locate entity chunks in text and classify their entity types. Originally a structured prediction task, NER has since been formulated as a task of sequential token labeling, much like text chunking and part-of-speech tagging. With the ability to compute representations of past and future context respectively for each token, bidirectional LSTM (Bi-LSTM) has proved a robust building block for sequence-labeling NER (Huang et al., 2015; Ma & Hovy, 2016; Chiu & Nichols, 2016). However, it has been predominantly used as a black box; research directed to understanding how the model learns to tackle the task is minimal.

In this work, we decouple sequence-labeling NER into the entity chunking and entity typing subtasks, and seek insight into what patterns LSTM learns to capture or has difficulties capturing. We propose the use of a fast and effective parallel self-attention mechanism alongside Bi-LSTM. Unlike traditional attention mechanisms used for tasks such as machine translation (Luong et al., 2015) and sentence classification (Conneau et al., 2017; Lin et al., 2017), our self-attentive Bi-LSTM uses the hidden state of each token as its own query vector and computes context vectors for all tokens in parallel. For both subtasks, we then find important global patterns that cross past and future context, and in particular discover the way multi-chunk entities are handled. Furthermore, we discover that the theoretical limitations of traditional Bi-LSTMs harms performance on the task, and hence propose using a cross construction of deep Bi-LSTMs. As a result, with these cross structures, both self-attentive Bi-LSTM and cross-Bi-LSTM achieve new state-of-the-art results on sequence-labeling NER.

In Section 3, the normal Bi-LSTM-CNN model is formulated. Section 4 details the computation of the parallel self-attention mechanism. Section 5 presents the empirical results and detailed analyses of the models, with a particular focus on patterns captured for $\{B, I, E\}$ labels. Finally in Section 6, cross-Bi-LSTM-CNN is formulated and evaluated on a theoretical basis. Our contribution is threefold:

- We provide insightful understanding of how a sequence-labeling model tackles NER and the difficulties it faces;
- We propose a fast and effective self-attention mechanism for sequence labeling;

- We propose using cross-Bi-LSTM-CNN for sequence-labeling NER with theoretically-grounded improvements.

2 Related Work

Many have attempted tackling the NER task with LSTM-based sequence encoders (Huang et al., 2015; Ma & Hovy, 2016; Chiu & Nichols, 2016; Lample et al., 2016). Among these, the most similar to the proposed Bi-LSTM-CNN is the model proposed by Chiu & Nichols (2016). In contrast to previous work, Chiu & Nichols (2016) stack multiple layers of LSTM cells per direction, and also use a CNN to compute character-level word vectors alongside pre-trained word vectors. We largely follow their work in constructing the Bi-LSTM-CNN, including the selection of raw features, the CNN, and the multi-layer Bi-LSTM. The subtle difference is that they send the output of each direction through separate affine-softmax classifiers and then sum their probabilities, effectively forming an ensemble of forward and backward LSTM-CNNs. Another difference is that they focus on proposing a new representation of external lexicon features, which we do not make use of in this work.

The modeling of global context for sequential-labeling NER has been accomplished using traditional models with intensive feature engineering and conditional random fields (CRF). Ratinov & Roth (2009) build the Illinois NER tagger with feature-based perceptrons. In their analysis, the usefulness of Viterbi decoding is minimal, as class transition patterns only occur in small chunks and greedy decoding can handle them comparatively well. On the other hand, recent research on LSTM or CNN-based encoders report empirical improvements brought by CRF (Huang et al., 2015; Ma & Hovy, 2016; Lample et al., 2016; Strubell et al., 2017), as it discourages illegal predictions by explicitly modeling class transition probabilities. In contrast, the cross structures of self-attention and cross-Bi-LSTM studied in this work provide for the direct capture of global patterns and extraction of better features to improve class observation likelihoods.

Various attention mechanisms have been proposed and shown success in natural language tasks. They lighten the LSTM's burden of compressing all relevant information into a single hidden state by consulting past memory. For seq2seq models, attention has been used for current decoder hidden states (Luong et al., 2015). For models computing sentence representations, trainable weights are used for self-attention (Conneau et al., 2017; Lin et al., 2017). In this work, we propose using a token-level parallel self-attention mechanism for sequential token-labeling and show that it enables the model to capture cross interactions between past and future contexts.

3 Bi-LSTM-CNN for Sequence Labeling

3.1 CNN and Word Features

All models in our experiments use the same set of raw features: word embedding, word capitalization pattern type, character embedding, and character type.

For character embedding, 25d vectors are randomly initialized and trained end-to-end with the model. Appended to these are 4d one-hot character-type features indicating whether a character is uppercase, lowercase, digit, or punctuation (Chiu & Nichols, 2016). In addition, an unknown character vector and a padding character vector are also trained. We unify the word token length to 20 by truncation and padding. The resulting 20-by-(25+4) feature map of each token are applied to a character-trigram CNN with 20 kernels per length 1 to 3 and max-over-time pooling to compute a 60d character-based word vector (Kim et al., 2016; Chiu & Nichols, 2016; Ma & Hovy, 2016).

For word embedding, pre-trained 300d GloVe word vectors (Pennington et al., 2014) are used without further tuning. In addition, 4d one-hot word capitalization features indicate whether a word is uppercase, upper-initial, lowercase, or mixed-caps (Collobert et al., 2011; Chiu & Nichols, 2016).

Throughout this paper, we use X to denote the n-by-d_x matrix of raw sequence features, with n denoting the number of word tokens in a sentence and $d_x = 60 + 300 + 4$.

3.2 Deep Bi-LSTM

Given a sequence of input feature vectors $x_1, x_2, \ldots, x_T \in R^{d1}$, an LSTM cell computes a sequence of hidden feature vectors $h_1, h_2, \ldots, h_T \in R^{d2}$ by

$$\begin{aligned} g_t &= \tanh(W_g x_t + V_g h_{t-1} + b_g) \\ i_t &= \sigma(W_i x_t + V_i h_{t-1} + b_i) \\ f_t &= \sigma(W_f x_t + V_f h_{t-1} + b_f) \\ o_t &= \sigma(W_o x_t + V_o h_{t-1} + b_o) \\ c_t &= g_t \odot i_t + c_{t-1} \odot f_t \\ h_t &= \tanh(c_t) \odot o_t, \end{aligned}$$

where $h_0, c_0 \in R^{d2}$ are zero vectors, $W_g, W_i, W_f, W_o \in R^{d2 \times d1}$, $V_g, V_i, V_f, V_o \in R^{d2 \times d2}$, $b_g, b_i, b_f, b_o \in R^{d2}$ are trainable weight matrices and biases, $\tanh$ denotes hyperbolic tangent, σ denotes sigmoid function, and $\odot$ denotes element-wise multiplication.

Bidirectional LSTMs (Bi-LSTMs) are used to capture the future and the past for each time step. Following Chiu & Nichols (2016), 4 distinct LSTM cells – two in each direction – are stacked to capture higher level representations:

$$\begin{aligned} \overrightarrow{H} &= \overrightarrow{LSTM}_2(\overrightarrow{LSTM}_1(X)) \\ \overleftarrow{H} &= \overleftarrow{LSTM}_4(\overleftarrow{LSTM}_3(X)) \\ H &= \overrightarrow{H} \mid\mid \overleftarrow{H}, \end{aligned}$$

where $\overrightarrow{LSTM}_i, \overleftarrow{LSTM}_i$ denote applying LSTM cell i in forward, backward order, $\overrightarrow{H}, \overleftarrow{H}$ denote the resulting feature matrices of the stacked application, and $||$ denotes row-wise concatenation. In all our experiments, 100d LSTM cells are used, so $H \in R^{n \times d_h}$ and $d_h = 200$.

3.3 Affine-Softmax and Chunk Labels

Finally, suppose there are d_p token classes, the probability of each of which is given by the composition of affine and softmax transformations:

$$\begin{aligned} s_t &= H_t W_p + b_t \\ p_{ti} &= \frac{e^{s_{ti}}}{\sum_{j=1}^{d_p} e^{s_{tj}}}, \end{aligned}$$

where H_t is the t^{th} row of H, $W_p \in R^{d_h \times d_p}$, $b \in R^{d_p}$ are a trainable weight matrix and bias, and s_{ti} and s_{tj} are the i-th and j-th elements of s_t.

Following Chiu & Nichols (2016), we use the 5 chunk labels O, S, B, I, and E to denote if a word token is $\{O\}$utside any entities, the $\{S\}$ole token of an entity, the $\{B\}$eginning token of a multi-token entity, $\{I\}$n the middle of a multi-token entity, or the $\{E\}$nding token of a multi-token entity. Hence when there are P types of named entities, the actual number of token classes $d_p = P \times 4 + 1$ for sequence labeling NER.

4 Self-Attentive Bi-LSTM-CNN for Sequence Labeling

4.1 Parallel Multi-Head Self-Attention

We propose using a token-level self-attention mechanism (Figure 1) that is computed after the auto-regressive Bi-LSTM in Section 3.2. This has two benefits over traditional auto-regressive attention, which wraps stacked LSTM cells to look at past tokens at each time step for each direction of Bi-LSTM. First, it allows each token to look at both past and future sequences simultaneously with one combined hidden state of past and future, thus capturing cross interactions between the two contexts. And secondly, since all time steps run in parallel with matrix computations, it introduces little computation time overhead.

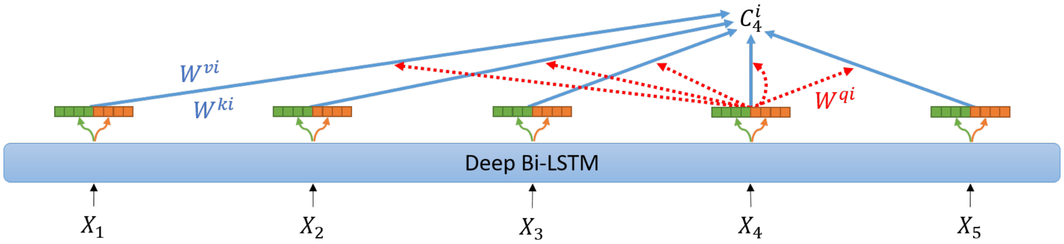


Figure 1: Computation of i-th attention head of 4th word token

Table 1: Dataset Statistics

Split	#Tokens	#Entities	Entity type: names	Entity type: values
Train Validate Test	1,088,503 147,724 152,728	81,828 11,066 11,257	PERSON, NORP, FAC, ORG, GPE, LOC, PRODUCT, EVENT, WORK_OF_ART, LAW, LANGUAGE	DATE, TIME, PERCENT, MONEY, QUANTITY, ORDINAL, CARDINAL
Total	1,388,955	104,151		

Specifically, given the hidden features H of a whole sequence, we project each hidden state to different subspaces, depending on whether it is used as the $\{$q$\}$uery vector to consult other hidden states for each word token, the $\{$k$\}$ey vector to compute its dot-similarities with incoming queries, or the $\{$v$\}$alue vector to be weighted and actually convey information to the querying token. Moreover, as different aspects of a task can call for different attention, multiple "attentions" running in parallel are used, i.e., multi-head attention (Vaswani et al., 2017).

Formally, let m be the number of attention heads and d_c be the subspace dimension. For each head $i \in \{1..m\}$, the attention weight matrix and context matrix are computed by

$$\alpha^i = \sigma\left(\frac{HW^{qi}(HW^{ki})^T}{\sqrt{d_c}}\right)$$

$$C^i = \alpha^i HW^{vi},$$

where $W^{qi}, W^{ki}, W^{vi} \in R^{d_h \times d_c}$ are trainable projection matrices and σ performs softmax along the second dimension. Each row of the resulting $\alpha^1, \alpha^2, \ldots, \alpha^m \in R^{n\times n}$ contains the attention weights of a token to its context, and each row of $C^1, C^2, \ldots, C^m \in R^{n\times d_c}$ is its context vector. Since $H = \overrightarrow{H} \mid\mid \overleftarrow{H}$, the computation of α^i and C^i models the cross interaction between past and future.

4.2 Affine-Softmax with Multi-Head Context

Finally, for Bi-LSTM-CNN augmented with the attention mechanism, the hidden vector and context vectors of each token are considered together for classification:

$$s_t^c = (H_t||C_t^1||C_t^2||...||C_t^m)W_c + b_t$$

$$p_{ti}^c = \frac{e^{s_{ti}^c}}{\sum_{j=1}^{d_p} e^{s_{tj}^c}},$$

where $||$ denotes row-wise concatenation, C_t^i is the t-th row of C^i, and $W_c \in R^{(d_h+md_c)\times d_p}$ are trainable weight matrices. In all our experiments, we use $m = 5$ and $d_c = \frac{d_h}{5}$, so $W_c \in R^{2d_h \times d_p}$.

Table 2: Overall Results

Model	Validate			Test		
	Prec.	Recall	F1	Prec.	Recall	F1
Finkel & Manning (2009)	-	-	-	84.04	80.86	82.42
Ratinov & Roth (2009)	-	-	-	82.00	84.95	83.45
Passos et al. (2014)	-	-	-	-	-	82.24
Durrett & Klein (2014)	-	-	-	85.22	82.89	84.04
Chiu & Nichols (2016)	-	-	-	86.04	86.53	86.28 (±0.26)
Strubell et al. (2017)	-	-	-	-	-	86.84 (±0.19)
Li et al. (2017)	85.5	84.7	85.08	88.0	86.5	87.21
Bi-LSTM-CNN	86.64	86.06	86.35	88.00	87.12	87.56 (±0.07)
Bi-LSTM-CNN +ATT	**87.22**	**86.69**	**86.95**	**88.79**	**87.81**	**88.29** (±0.20)

5 Evaluation and Analysis

5.1 Dataset

We conduct experiments on the challenging OntoNotes 5.0 English NER corpus (Hovy et al., 2006; Pradhan et al., 2013). OntoNotes is an ambitious project that collects large corpora from diverse sources and provides multi-layer annotations for joint research on constituency parsing, semantic role labeling, coreference resolution, and NER. The data sources include newswires, web, broadcast news, broadcast conversations, magazines, and telephone conversations. Some are transcriptions of talk shows and some are translated from Chinese or Arabic. Such diversity and noisiness requires that models are robust and able to capture a multitude of linguistic patterns.

Table 1 summarizes the dataset statistics. Following previous lines of research, we use the standard split provided by Pradhan et al. (2013), excluding the New Testament corpus as it contains no entity annotations. Despite this million-token corpus with over 100K annotated entities, previous work has struggled to reach state-of-the-art NER results on the dataset. This is due partly to the fact that there are 18 types of entities to be classified. Eleven of these are classes of general names, with *NORP* including nationalities such as *American*, *FAC* including facilities such as *The White House*, and *WORK_OF_ART* including titles of books, songs, and so on. Moreover, various forms of values of the seven numerical classes must also be identified.

5.2 Implementation Details

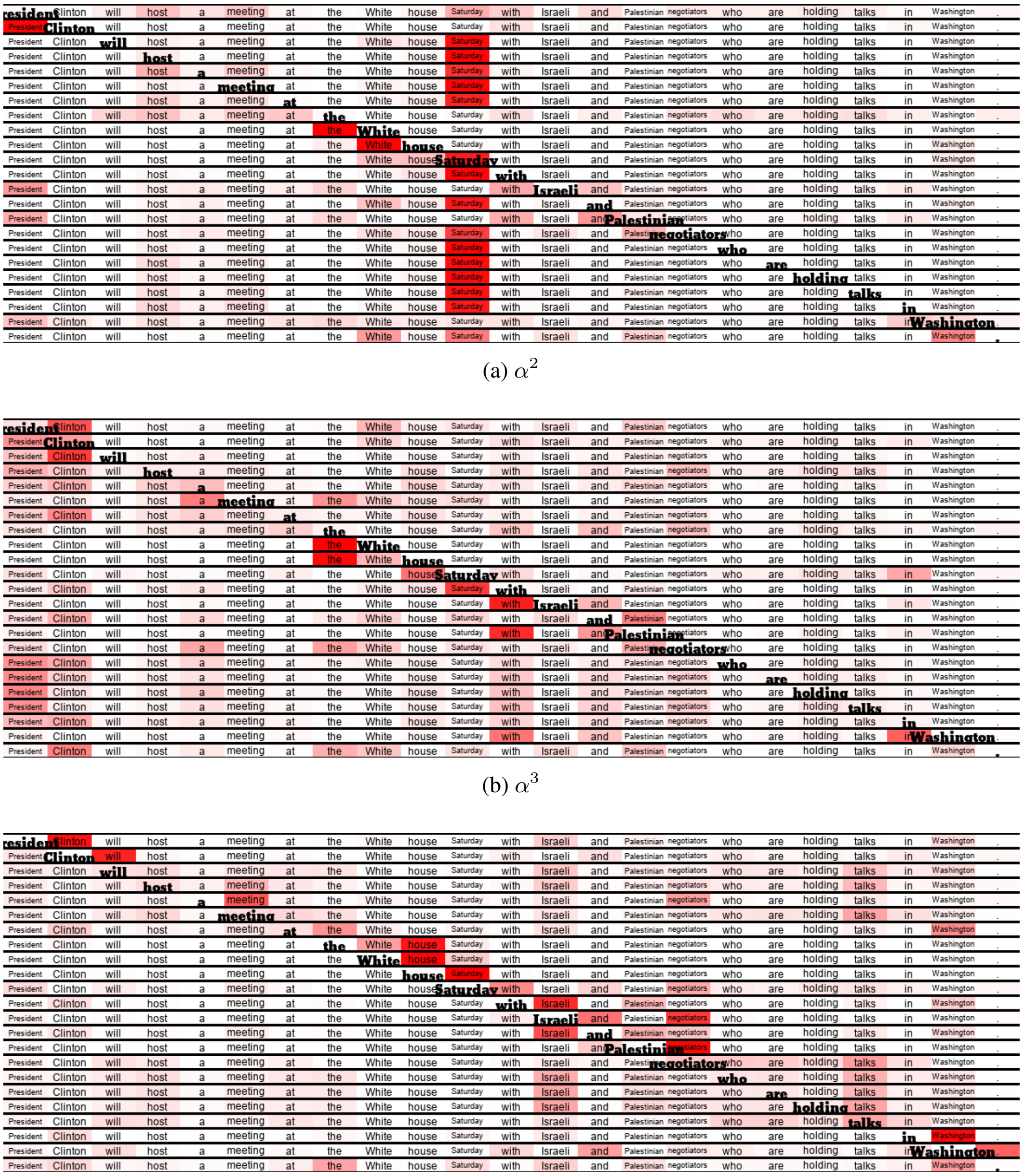
The hyperparameters of our models were given in Sections 3 and 4. When training the models, we minimized per-token cross-entropy loss with the Nadam optimizer (Dozat, 2016). In addition, we randomly dropped 35% hidden features (dropout) and upscaled the same amount during training. Following previous lines of work, we evaluated NER performance with the per-entity F1 score. The tokens for an entity were all to be classified correctly to count as a correct prediction; otherwise it was counted as either a false positive prediction or a false negative non-prediction. We stopped training when the validation F1 had not improved for 20 epochs. All models were initialized and trained 5 times; we report the mean precision, recall, and F1 scores (%) of the experiments. Validation scores are also reported for future research on this task.

5.3 Overall Results

Table 2 shows the overall results of our models against notable previous work. It can be seen that simple LSTM-based sequence encoders already beat the previous best results without using external lexicons (Chiu & Nichols, 2016), document-level context (Strubell et al., 2017), or constituency parsers (Li et al., 2017). Furthermore, with the proposed parallel self-attention mechanism (ATT), we achieve a new state-of-the-art result (**88.29** F1) with a clear margin over past systems. More importantly, the attention mechanism allows us to conduct insightful analyses in the following sections, yielding important understanding of how Bi-LSTM learns or has difficulty tackling the different sequence-labeling NER subtasks: entity chunking and entity typing.

Table 3: Chunking Results

	HC^{all}	H	C^{all}	C^1	C^2	C^3	C^4	C^5	$NativeH$
O	99.05	-1.68	0.75	0.95	-1.67	-45.57	-0.81	-35.46	-0.03
S	93.74	2.69	-91.02	-90.56	-90.88	-25.61	-86.25	-84.32	0.13
B	90.99	1.21	-52.26	-90.78	-88.08	-90.88	**-12.21**	-87.45	**-0.63**
I	90.09	**-28.18**	**-3.80**	-87.93	**-60.56**	**-50.19**	**-57.19**	-79.63	**-0.41**
E	93.23	2.00	-71.50	-93.12	**-36.45**	**-39.19**	-91.90	-90.83	**-0.38**



(a) α^2

(b) α^3

(c) α^4

Figure 2: Attention heat maps for "...a meeting at the White house Saturday..."

5.4 Entity Chunking

We decouple the entity chunking task from sequence-labeling NER. Specifically, for a sentence such as $\{Barack\ Obama\ moves\ out\ of\ the\ White\ House\ .\}$, the task is to correctly label each token as $\{Barack_B\ Obama_E\ moves_O\ out_O\ of_O\ the_B\ White_I\ House_E\ ._O\}$.

5.4.1 Quantitative Analysis

Table 3 shows the performance of different setups on validation data. We take the pre-trained models from Table 2 without re-training for this subtask. $\{O, S, B, I, E\}$ are the chunk classes. The column of HC^{all} lists the performance of the full Bi-LSTM-CNN+ATT model on each chunk class, where C^{all} stands for $C^1, \ldots, C^5$. Other columns list the performance of other setups compared to the full model. Columns H to C^5 are when the full model is deprived of all other information by zeroing all other vectors for the affine-softmax classification layer in testing time, except for those specified by the column header. $NativeH$ is the native Bi-LSTM-CNN trained without attention. The figures shown in the table are the per-token recalls for each chunk class, which tells if a part of the model is responsible for signaling the whole model to predict the class.

Looking at the three columns on the left, the first thing we discover is that Bi-LSTM-CNN+ATT designates the task of predicting $\{I\}$ to the attention mechanism. The model performance on tokens $\{I\}$n the middle of an entity significantly degrades (**-28.18**) in the absence of global context C^{all}, when token hidden state H is left alone. On the other hand, without the information on the token itself, it is clear that the model strongly favors predicting I (**-3.80**) given its global context C^{all}.

Taking this one step further and zeroing out all other vectors except for each attention head, the roles of context for entity chunking become even clearer. C^2 and C^3 send strong signals (**-36.45,-39.19**) on entity chunk $\{E\}$nding to the model, plus weak signals (**-60.56,-50.19**) on entity chunk $\{I\}$nside, while C^4 sends a strong signal (**-12.21**) on entity chunk $\{B\}$eginning plus weak signals (**-57.19**) on $\{I\}$nside. When all these heads fire simultaneously, the model produces a strong signal to $\{I\}$.

However, $NativeH$ – Bi-LSTM-CNN trained without attention – underperforms in chunk labels $\{B\}$ (**-0.63**), $\{I\}$ (**-0.41**), $\{E\}$ (**-0.38**) in comparison to HC^{all}, the model trained with ATT. This suggests that entity chunking is indeed a crucial aspect in sequence-labeling NER, and that it is difficult for pure LSTM encoders to compress all necessary information in each hidden state to correctly label all the tokens of a multi-token entity.

5.4.2 Qualitative Analysis

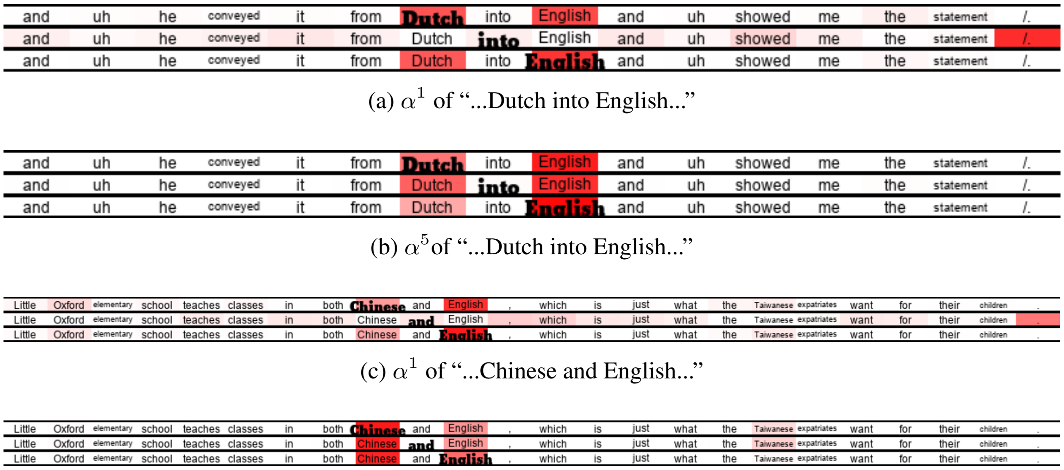
Aside from knowing that entity chunking is a crucial, challenging aspect in sequence-labeling NER for Bi-LSTM, one remaining question is how exactly the encoder is attempting to properly classify the $\{B\}$egin, $\{I\}$nside, and $\{E\}$nd of a multi-token entity. To shed light on this question, we visualize samples from validation data and discover consistent patterns in the attention weight heat maps across sentences and entities.

Figure 2 shows one of the samples, where the attention weights $\alpha^2, \alpha^3, \alpha^4$ of a sentence containing $the_B\ White_I\ house_E$ are visualized. The full Bi-LSTM-CNN+ATT (HC^{all}) classifies the tokens correctly, but when in the absence of the context vectors (H), the predictions become $the_B\ White_S\ house_E$. For Bi-LSTM-CNN trained without attention at all ($NativeH$), the predictions are $the_O\ White_S\ house_O$. Each row of the matrix shows the attention weight distribution for the diagonal token in bold font.

We observe that α^2 and especially α^3 have a tendency to focus on the previous tokens: the diagonal shifted left. In contrast, α^4 tends to look at the immediate following tokens: the diagonal shifted right. By looking for *previous* tokens that belong to the same entity chunk and finding some, an attention head, via its context vector, can signal to the model that the token spoken of might be the $\{E\}$nding token or $\{I\}$nside token. The same is true for an attention head looking at *next* tokens, but this time signaling for $\{B\}$egin and $\{I\}$nside. This also dictates that both signals need to be weaker for $\{I\}$ but stronger when combined. This behavior can be observed throughout the heat maps of $\alpha^2, \alpha^3, \alpha^4$. In particular for $the\ White\ house$, C^{all} predicts $the_B\ White_I\ house_O$ as $Saturday$ is wrongly focused by α^4 for $house$.

Table 4: Notable Typing Results

	HC^{all}	H	C^{all}	C^1	C^2	C^3	C^4	C^5	$NativeH$
FAC	63.84	-36.17	-2.83	-62.90	-39.00	-29.25	-13.84	-53.46	**-4.41**
LOC	73.47	-15.56	-42.09	-73.47	-67.35	-67.60	-41.33	-60.97	**-5.36**
LAW	54.03	-25.80	-1.61	-54.03	-11.29	-11.29	-24.19	-12.90	**3.23**
LAN	63.64	-27.28	-60.61	-63.64	-63.64	-63.64	-60.61	**27.27**	**-9.09**



(a) α^1 of "...Dutch into English..."

(b) $\alpha^5$of "...Dutch into English..."

(c) α^1 of "...Chinese and English..."

(d) α^5 of "...Chinese and English..."

Figure 3: Attention heat maps of "...Dutch into English..." and "...Chinese and English..."

From Table 3, we already know that $NativeH$ has some difficulties in handling multi-token entities, being more inclined to predict $\{S\}$ingle-token entities, and that HC^{all} mitigates this problem by delegating work to C^{all}, especially by relying on the latter to signal for $\{I\}$n tokens. The heat maps further tell the story of how the related labels $\{B, I, E\}$ are handled collectively. In addition, this also suggests that modeling interactions between future and past contexts is crucial for sequence-labeling NER and motivates the use of a deep cross-Bi-LSTM encoder in Section 6.

5.5 Entity Typing

When the entity chunking task is decoupled from sequence-labeling NER, the remaining entity typing task requires a model to label $\{Barack\ Obama\ moves\ out\ of\ the\ White\ House\ .\}$ as $\{Barack_{PERSON}\ Obama_{PERSON}\ moves_{NONE}\ out_{NONE}\ of_{NONE}\ the_{FAC}\ White_{FAC}\ House_{FAC}\ ._{NONE}\}$. Table 4 shows the entity classes for which HC^{all} yields notably different performance ($> 2\%$) from that of $NativeH$. Of particular interest is C^5's strong signal (**27.27**) for LAN (language) in comparison to the $NativeH$'s struggles (**-9.09**) on this class without attention.

Qualitatively, we study the two sentences shown in Figure 3, containing $Dutch_{LAN}\ into_{NONE}\ English_{LAN}$ and $Chinese_{LAN}\ and_{NONE}\ English_{LAN}$. HC^{all} classifies the tokens correctly, but both H and $NativeH$ wrongly predict $Dutch_{NORP}\ into_{NONE}\ English_{LAN}$ and $Chinese_{NORP}\ and_{NONE}\ English_{LAN}$. Here $NORP$ stands for nationality, meaning that both models without attention wrongly judge that $Dutch$ and $Chinese$ here refer to people from these countries.

With attention, in Figure 3, we see that α^1 attends to $Dutch$ and $English$ at the same time for the two tokens and attends to $Chinese$ and $English$ at the same time for the other two. On the other hand, α^5 focuses on all possible LAN tokens, including a small mis-attention to $Taiwanese$ in the second sentence, which is actually a $NORP$ in this case. These attention weights signify that the model learns a pattern of cross interaction between entities: when two ambiguous entities of $NORP$, LAN occur together in the same context, the model predicts both as LAN.

6 Cross Structures for Sequence Labeling

6.1 Theoretical Limitation of Bi-LSTM

In Section 4.1, we briefly mentioned that the computation of attention weights α^i and context features C^i models the cross interaction between past and future. Mathematically, since $H = \overrightarrow{H} \mid\mid \overleftarrow{H}$, the computation of attention scores can be rewritten as

$$HW^{qi}(HW^{ki})^T = (\overrightarrow{H} \mid\mid \overleftarrow{H})(W^{qi}W^{ki^T})(\overrightarrow{H} \mid\mid \overleftarrow{H})^T.$$

The un-shifted covariance matrix of the projected $(\overrightarrow{H} \mid\mid \overleftarrow{H})$ thus computes the interaction between past context and future context for each token, capturing cross-context patterns that the deep Bi-LSTM-CNN specified in Section 3 cannot. The consequence of this inability has been empirically shown in Section 5. Here, we further consider the following four simple phrases that form an *XOR*:

$$\{Key\ and\ Peele\}_{WOA}; \{You\ and\ I\}_{WOA}; \{Key\ and\ I\}; \{You\ and\ Peele\}$$

where WOA stands for $WORK\_OF\_ART$. The first two phrases are respectively a show title and a song title. The other two are not entities, where the last one actually occurs in an interview with Keegan-Michael Key. Suppose the phrases themselves are the only available context for the classification of and. Then the Bi-LSTM-CNN cannot capture good enough features to classify and correctly simultaneously for the four cases, even if they are the training data, no matter how many LSTM cells are stacked. The key is that given the same half-context of past or future, and is sometimes $\{WOA : I\}$ but sometimes $\{NONE : O\}$. It is only when patterns that cross past and future are captured that the model is able to decide the correct label.

6.2 Cross-Bi-LSTM-CNN for Sequence Labeling

Motivated by the limitation of the conventional Bi-LSTM-CNN for sequence labeling, we propose the use of Cross-Bi-LSTM-CNN by changing the deep structure in Section 3.2 to

$$H^1 = \overrightarrow{LSTM}_1(X) \mid\mid \overleftarrow{LSTM}_3(X)$$

$$H = \overrightarrow{LSTM}_2(H^1) \mid\mid \overleftarrow{LSTM}_4(H^1).$$

Note that when computing sentence embeddings for tasks such as sentence classification, both directions of a normal Bi-LSTM look at the whole sentence. However, when computing hidden node features for sequence labeling, each direction of a normal Bi-LSTM looks at only half of the sentence. Cross-Bi-LSTM remedies this problem by interleaving the hidden features between LSTM layers. The output of the first layers of both directions are sent to the second layers of both directions, allowing higher layers to capture interactions between past and future contexts for each token. Empirically, we experiment with cross construction 5 times and find it further improves the performance of Bi-LSTM-CNN from 87.56 (±0.07) to 88.09 (±0.16).

7 Conclusion

In this paper, we have decoupled named entity recognition into entity chunking and entity typing and demonstrated how sequence-labeling models can learn to handle each of these two subtasks. By using a fast parallel self-attention mechanism, we have discovered how the beginning and ending of a multi-token entity is determined and how they are jointly correlated to locate the inside tokens. Further, through our quantitative and qualitative analyses for both chunking and typing, we have shown that it is crucial to capture global patterns that cross both sides of a token. We demonstrate the theoretical limitation of the conventional deep Bi-LSTM-CNN used in sequence labeling tasks. In addition to the interpretability of the proposed parallel self-attention, it is shown that it constitutes a way to correlate past and future contexts. We have also provided deep cross-Bi-LSTM-CNN as another way to extract global context features. With their respective cross structures, both self-attentive Bi-LSTM and cross-Bi-LSTM achieve new state-of-the-art results on sequence-labeling NER.

REFERENCES

Jason Chiu and Eric Nichols. Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4:357–370, 2016. ISSN 2307-387X. URL https://transacl.org/ojs/index.php/tacl/article/view/792.

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

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 670–680, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D17-1070.

Timothy Dozat. Incorporating Nesterov momentum into Adam. In *Proceedings of ICLR 2016 Workshop*, 2016.

Greg Durrett and Dan Klein. A joint model for entity analysis: Coreference, typing, and linking. *Transactions of the Association for Computational Linguistics*, 2:477–490, 2014. URL http://aclweb.org/anthology/Q14-1037.

Jenny Rose Finkel and Christopher D. Manning. Joint parsing and named entity recognition. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 326–334, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/N/N09/N09-1037.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. Ontonotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pp. 57–60, New York City, USA, June 2006. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/N/N06/N06-2015.

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

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. Character-aware neural language models. In *AAAI*, pp. 2741–2749, 2016.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 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*, pp. 260–270, San Diego, California, June 2016. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/N16-1030.

Peng-Hsuan Li, Ruo-Ping Dong, Yu-Siang Wang, Ju-Chieh Chou, and Wei-Yun Ma. Leveraging linguistic structures for named entity recognition with bidirectional recursive neural networks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2664–2669, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D17-1282.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. In *International Conference on Learning Representations (ICLR)*, 2017.

Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics. URL http://aclweb.org/anthology/D15-1166.

Xuezhe Ma and Eduard Hovy. 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)*, pp. 1064–1074, Berlin, Germany, August 2016. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P16-1101`.

Alexandre Passos, Vineet Kumar, and Andrew McCallum. Lexicon infused phrase embeddings for named entity resolution. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pp. 78–86. Association for Computational Linguistics, 2014. doi: 10.3115/v1/W14-1609. URL `http://www.aclweb.org/anthology/W14-1609`.

Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/D14-1162`.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. Towards robust linguistic analysis using OntoNotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pp. 143–152, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/W13-3516`.

Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pp. 147–155. Association for Computational Linguistics, 2009. URL `http://www.aclweb.org/anthology/W09-1119`.

Emma Strubell, Patrick Verga, David Belanger, and Andrew McCallum. Fast and accurate entity recognition with iterated dilated convolutions. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2670–2680, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/D17-1283`.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 5998–6008. Curran Associates, Inc., 2017. URL `http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf`.