Pattern Recognition 110 (2021) 107636

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Position-aware self-attention based neural sequence labeling

Wei Wei[a,1,*], Zanbo Wang[a], Xianling Mao[b], Guangyou Zhou[c], Pan Zhou[d], Sheng Jiang[a]

[a] School of Computer Science and Technology, Huazhong University of Science and Technology (HUST), Wuhan, Hubei, 430074, China
[b] School of Computer, Beijing Institute of Technology, Beijing 100081, China
[c] School of Computer, Central China Normal University, Wuhan, Hubei, 430000 China
[d] School of Cyber Science and Engineering, Huazhong University of Science and Technology (HUST), Wuhan, Hubei, 430074, China

A R T I C L E I N F O

Article history:
Received 24 January 2020
Revised 30 April 2020
Accepted 6 September 2020
Available online 7 September 2020

Keywords:
Equence labeling
Self-attention
Discrete context dependency

A B S T R A C T

Sequence labeling is a fundamental task in natural language processing and has been widely studied. Recently, RNN-based sequence labeling models have increasingly gained attentions. Despite superior performance achieved by learning the long short-term (*i.e., successive*) dependencies, the way of sequentially processing inputs might limit the ability to capture the non-continuous relations over tokens within a sentence. To tackle the problem, we focus on how to effectively model *successive* and *discrete* dependencies of each token for enhancing the sequence labeling performance. Specifically, we propose an innovative attention-based model (called position-aware self-attention, *i.e.,* PSA) as well as a well-designed self-attentional context fusion layer within a neural network architecture, to explore the positional information of an input sequence for capturing the latent relations among tokens. Extensive experiments on three classical tasks in *sequence labeling* domain, *i.e., part-of-speech* (**POS**) *tagging, named entity recognition* (**NER**) and *phrase chunking*, demonstrate our proposed model outperforms the state-of-the-arts without any external knowledge, in terms of various metrics.

1. Introduction

Sequence labeling, named SL, is one of *pattern recognition* task in the filed of natural language processing (NLP) and machine learning (ML), which aims to assign a categorical label to each element of a sequence of observed values, such as *part-of-speech* (POS) *tagging* [24], *chunking* [22] and *named entity recognition* (NER) [17] and *etc.* It plays a pivotal role in *natural language understanding* (NLU) and significantly beneficial for a variety of *downstream* applications, *e.g., syntactic parsing, relation extraction* and *entity coreference resolution* and *etc.*

Conventional sequence labeling approaches are usually on the basis of classical machine learning technologies, such as *Hidden Markov Models* (HMM) [3] and *Conditional Random Fields* (CRF) [16], which heavily rely on hand-crafted features (*e.g.,* with/without capitalized word) or language-specific resources (*e.g.,* gazetteers), making it difficult to apply them to new language-related tasks or domains. With advances in deep learning, many research efforts have been dedicated to enhancing SL [24] by automatically extracting features via different types of neural networks (NNs), where various characteristics of word information are encoded in distributed representations for inputs [7] and the sentence-level context representations are learned when end-to-end training.

Recently, Recurrent Neural Network (RNN) together with its variants, *e.g.,* long short-term memory (LSTM) or gated recurrent unit (GRU), have shown great success in modeling sequential data [46]. Therefore, many researches have devoted to research on RNN based architectures for SL, such as BiLSTM-CNN [5], LSTM-CRF [15,19], LSTM-CNN-CRF [24] and *etc.* Despite superior performance achieved, these models have limitations under the fact that RNNs recursively compose each word with its previous hidden state encoded with the entire history information, but the latent independent relations between each pair of words are not well managed. The sequential way to process the inputs only focuses on modeling the long-range *successive* context dependencies, while neglecting the *discrete* context patterns.[2]

Discrete context dependency plays a significant role in sequence labeling tasks. Generally, for a given word, its label not only

* Corresponding author.
E-mail addresses: weiw@hust.edu.cn (W. Wei), jwt@hust.edu.cn (Z. Wang), maoxl@bit.edu.cn (X. Mao), gyzhou@mail.ccnu.edu.cn (G. Zhou).

[1] This work was supported in part by the National Natural Science Foundation of China under Grant No. 61602197 and Grant No. 61772076, Grant No. 61972448, Grant No. L1924068 and in part by Equipment Pre-Research Fund for The 13th Five-year Plan under Grant No. 41412050801.

[2] In the case study part of Section 5, we present multiple error cases to help explain and prove that the RNN models may cause insufficient modelling of discrete context dependency.

https://doi.org/10.1016/j.patcog.2020.107636

Traditional Industries[NNP] Inc. said it is seeking new financing.

A. Manual typewriters, B[LS]. black snapshots, C. radio adventure shows.

Fig. 1. Example: the impacts of discrete context dependencies.

depends on its own semantic information and neighbor contexts, but may also rely on the separate word information within the same sequences, which would significantly affect the accuracy of labeling. Without loss of generality, we take the part-of-speech (POS) tagging task as example, as shown in Fig. 1, the part-of-speech tag of word "Industries" in *Sentence-1* primarily depends on word "it", and thus should label with NNP, which refers to *singular proper noun*. However, if such discrete context dependency is not well modeled, "Industries" may tend to be labeled with *plural proper noun* (NNPS) mistakenly, since a word ending with "-s" and more so "-es" is more likely labeled with NNPS. Similar to *Sentence-2*, assigning the part-of-speech tag *list item marker* (LS) to word "B" should take account of word "A" and "C", where these three constitute a list. Therefore, it is essential to selectively choose the contexts that have strong impacts on the tag of the given word.

Many works demonstrate that self-attention is capable of effectively improving the performance of several NLP tasks such as machine translation, reading comprehension and semantic role labeling. This inspires us to introduce *self-attention* to explicitly model *position*-aware contexts of a given sequence. Although encoding *absolute* positions of the input sequence with attentions [43] has been proven the effectiveness, compared with injecting *absolute* position embedding into the initial representations, it is more intuitive to incorporate the positional information in a *relative* manner. Recently, Shaw et al. [34] present an alternative approach to take the account of the *relative* distance between sequence elements for representation. Nevertheless, their approaches only consider the relative position information independent of the sequence tokens while neglecting the interaction with the input representations. Hence, how to effectively exploit the position information with attentions for better modeling the context dependency is still an open problem.

In this paper, we propose a novel RNN neural architecture for sequence labeling tasks, which employs self-attention to implicitly encode position information to provide complementary context information on the basis of Bi-LSTM. Additionally, we further propose an extension of *standard* additive self-attention mechanism (named position-aware self-attention, PSA) to model the discrete context dependencies of the input sequence. Differ from previous works, PSA maintains a *variable-length* memory to explore position information in a more flexible manner for tackling the above mentioned problem. That is, it jointly exploits three different positional bias, *i.e., self-disabled mask* bias, *distance-aware Gaussian* bias and *token-specific position* bias, to induce the latent independent relations among tokens, which can effectively model the discrete context dependencies of given sequence. Additionally, we also develop a well-designed self-attentional context fusion layer with feature-wise gating mechanism to dynamically select useful information about discrete context dependency and also address the *self-disabled mask* bias problem. Specifically, it learns a parameter λ to adaptively combine the input and the output of the *position*-aware self-attention and then generate the context-aware representations of each token. The extensive experiments conducted on four classical benchmark datasets within the domain of *sequence labeling, i.e.,* the CoNLL 2003 NER, the WSJ portion of the Penn Treebank POS tagging, the CoNLL 2000 chunking and the OntoNotes 5.0 English NER, demonstrate that our proposed model achieves a significant improvement over the state-of-the-arts. The main contributions of this work are as follows.

- We identify the problem of modeling discrete context dependencies in sequence labeling tasks.
- We propose a novel *position*-aware self-attention to incorporate three different positional factors for exploring the *relative* position information among tokens; and also develop a well-designed self-attentional context fusion with feature-wise gating mechanism to provide complementary context information on the basis of Bi-LSTM for better modeling the discrete context dependencies over tokens.
- Extensive experiments on *part-of-speech* (**POS**) *tagging, named entity recognition* (**NER**) and *phrase chunking* tasks verify the effectiveness of our proposed model.

Roadmap. The remaining of the paper is organized as follows. In Section 2, we review the related work, and in Section 3 we presents a background on sequence labeling tasks, as well as a *Bi-LSTM-CRF* baseline model, followed with the proposed *position*-aware self-attention mechanism and self-attentional context fusion layer in Section 4. Section 5 presents the quantitative results on benchmark datasets, also includes an in-depth analysis, case study and wraps up discussion over the obtained results. Finally, Section 6 concludes the paper.

2. Related work

There exist three threads of related work regarding our proposed sequence labeling problem, namely, *sequence labeling, self-attention* and *position based attention*.

2.1. Sequence labeling

Sequence labeling is a category of fundamental tasks in natural language processing (NLP), *e.g.,* POS tagging, phrase chunking, named entity recognition (NER) and *etc.* Most of conventional high performance sequence labeling approaches are based on classical statistical machine learning models, such as HMM [31], CRFs [16,25], Support Vector Machine (SVM) [14], Perceptron [6], and *etc.*, where the well-designed features are required for training. Although the great success has been achieved by the traditional supervised learning based methods, these approaches require a lot of engineering skill and domain expertise to design handcrafted features.

With the rise of deep learning, many research efforts have been conducted on neural network based approaches to automatically learning the feature representation for SL tasks. The pioneering work is firstly proposed by Collobert et al. [7] to extract context-aware features using a simple feed-forward neural network with a fixed-size window, and generate the final labeled sequence through a CRF layer, which yields good performance in POS tagging, chunking, NER and *etc.* However, such window-based methods essentially follow a hypothesis, according to which the tags of an input word mainly depend on its neighboring words, while neglecting the global long-range contexts.

Hence, several variants of bidirectional recurrent neural networks, *e.g.,* Bidirectional Long Short-Term Memory (Bi-LSTM) and Bidirectional Gated Recurrent Unit (Bi-GRU), are proposed to encode long-range dependency features for the representation learning of each word, and thus achieve excellent performances. Huang et al. [13] initially employ a Bi-LSTM model to encode contextual representations of each word and then adopt a CRF model to jointly decode. Subsequently, the proposed Bi-LSTM-CRF architecture is widely used for various sequence labeling tasks. Lample et al. [17] and Ma et al. [24] both extend such model with an additional LSTM/CNN layer to encode character-level representations. Liu et al. [22] conduct a multi-task learning for sequence labeling by incorporating a character-aware neural language model. Zhang et al. [48] propose a multi-channel model to learn the tag dependency via a combination of word-level Bi-LSTM and tag LSTM. Besides, there also exist several Bi-GRU based sequence labeling models, *e.g.,* [44]. However, these RNN-based architectures are poor in modeling discrete context dependencies. In contrary, our proposed model is based on the Bi-LSTM-CRF architecture with self-attention mechanism to model the discrete position-aware dependencies for addressing the sequence labeling problem.

2.2. Attention mechanism

Self-Attention. Here, we mainly focus on reviewing self-attention based methods. Self-attention is a special case of the attention mechanism to flexibly capture both successive and discrete dependencies over a given sequence. Indeed, many studies have devoted to research on how to utilize self-attention mechanisms to improve the performance of several NLP tasks through aligning scores of different elements within a sequence, such as reading comprehension [4], textual entailment [21], sentiment analysis [21], machine translation [43], language understanding [35] and semantic role labeling [42]. Cheng et al. [4] extend the LSTM architecture with self-attention to enable *adaptive* memory usage during recurrence, which favors to several NLP tasks, ranging from *sentiment analysis* to *natural language inference*. Lin et al. [21] introduce a sentence embedding model with self-attention, in which a 2-dimensional matrix is utilized to represent the embedding and each row of the matrix attends on a different part of the sentence. The model is applied to *author profiling, sentiment analysis* and *textual entailment*, and yields a significant performance gain over other methods. Vaswani et al. [43] propose a RNN/CNN free self-attention network to construct a *sequence-to-sequence* (*i.e.,* seq2seq) model and achieve the state-of-the-arts in the neural machine translation (NMT) task. Shen et al. [35] employ self-attention to encode sentences and achieve great inference quality on a wide range of NLP tasks.

However, the purposes of these studies are different from the current work and thus will not be discussed in detail. The most related work is proposed by Tan et al. [42], where they propose a deep neural architecture with self-attention mechanism for *semantic role labeling* task and achieves the excellent performance, which inspire us to follow this line to apply self-attention to sequence labeling tasks for better learning the *word*-level context features and modeling the discrete dependencies over a given sequence.

Position based Attention. Attention mechanism has strong ability to model dependencies among tokens, but it cannot effectively make full use of the position information of the sequence in its structure. Vaswani et al. [43] propose a transformer model solely based on attention mechanism that achieves excellent performance for Neural Machine Translation (NMT) tasks, and they also point out the problem of neglecting the position information within attention in the existing methods. As such, they consider to inject position information using timing signal approach to encode absolute position, and then embed it into the representation of the input sequence in pre-processing progress with attentions. Following the success of Transformer, several subsequent studies using the Transformer architecture with the same strategy are proposed [42]. Show et al. [34] extend the self-attention mechanism to take into account the representations of the *relative* distances among sequence elements, and yields the substantial improvements in NMT task. Similarly, Sperber et al. [38] model the *relative* position information by strictly limit the scope of self-attention within their neighboring representations, which favors to the long-sequence acoustic modeling. Nevertheless, these approaches solely take account of the absolute or relative position information independent of sequence tokens while neglecting its interactions with their input presentations. In contrast, our proposed *position*-aware self-attention model explore the positional information of the given sequence in a more flexible manner, *i.e.,* mainly focusing on modeling of discrete context dependencies of that sequence.

3. Preliminary

Typically, sequence labeling can be treated as a set of independent classification tasks, which makes the optimal label for each member and then the global best set of labels is chosen for the given sequence at once. Suppose we have a sequence ($\hat{\mathbf{x}}$) composed of n tokens, *i.e.,* $\hat{\mathbf{x}} = [x_1, x_2, \ldots, x_n]^\top$, we aim to assign a tag to each member and output the corresponding globally best label sequence $\hat{\mathbf{y}} = [y_1, y_2, \ldots, y_n]^\top$. Many neural models are proposed for this task [17,24]. By following the success of the state-of-the-art neural network architecture, we briefly describe a *Bi-LSTM-CRF* model for this task, which often consists of three major stages:

Distributed Representation, represents words in low dimensional real-valued dense vectors, where each dimension represents a latent feature. Besides pre-trained word embeddings for the basic input, several studies also incorporate character-level representations for exploiting useful intra-word information (*e.g.,* prefix or suffix).

Context Encoder, captures the context dependencies and learns contextual representations for tag decoding. Traditional methods easily face the risk of gradient vanishing/exploding problem, and thus several variants of RNNs, *e.g.,* LSTMs [12], are widely employed to be the context encoder architecture for different sequence labeling tasks, owing to their promising performance on handling such problems. Therefore, here we briefly illustrate a special case of LSTM-CRF model, *i.e., Bi-directional* LSTM-CRF, which incorporate past/future contexts from both directions (forward/backward) to generate the hidden states of each word, and then jointly concatenate them to represent the *global* information of the entire sequence.

However, the sequential way to process the inputs of RNNs might weaken the sensitivity of modeling discrete context dependencies, since it recursively compose each word with its forward/backward hidden state that encodes the entire history/future information. As such, the latent relationship between each pair of words is not well extracted, which is closely related to the final prediction task. To this end, in this paper we propose a self-attentional context fusion layer to better capture the relations among tokens and help to model discrete context dependencies, via incorporating the complementary context information at different layers in our proposed neural architecture. We will detail it in the following sections, respectively.

Tag Decoder, employs a CRF layer to produce a sequence of tags corresponding to the input sequence. Typically, the correct label to each element of a given sequence often depends on the choices of nearby elements. As such, the correlations between labels of adjacent neighborhoods are usually considered for jointly decoding the best chain of labels for the entire sequence. Additionally, CRF model has been proven [16] to be powerful in learning the strong dependencies across output labels, thus it is usually employed to make the optimal label for each element of the input sequence. Specifically, let $\mathbf{Z} = [\hat{\mathbf{z}}_1, \hat{\mathbf{z}}_2, \ldots, \hat{\mathbf{z}}_n]^\top$ be the output of context encoder of the given sequence $\hat{\mathbf{x}}$, and thus the probability $\Pr(\hat{\mathbf{y}}|\hat{\mathbf{x}})$ of generating the whole label sequence $\hat{\mathbf{y}}$ with regard to $\mathbf{Z}$ is calculated by CRF model [22],

$$\Pr(\hat{\mathbf{y}}|\hat{\mathbf{x}}) = \frac{\prod_{j=1}^{n} \phi(y_{j-1}, y_j, \hat{\mathbf{z}}_j)}{\sum_{y' \in \mathbf{Y}(\mathbf{Z})} \prod_{j=1}^{n} \phi(y'_{j-1}, y'_j, \hat{\mathbf{z}}_j)}, \tag{1}$$

$$\phi(y_{j-1}, y_j, \hat{\mathbf{z}}_j) = \exp(\mathbf{W}_{y_{j-1}, y_j} \hat{\mathbf{z}}_j + b_{y_{j-1}, y_j}), \tag{2}$$

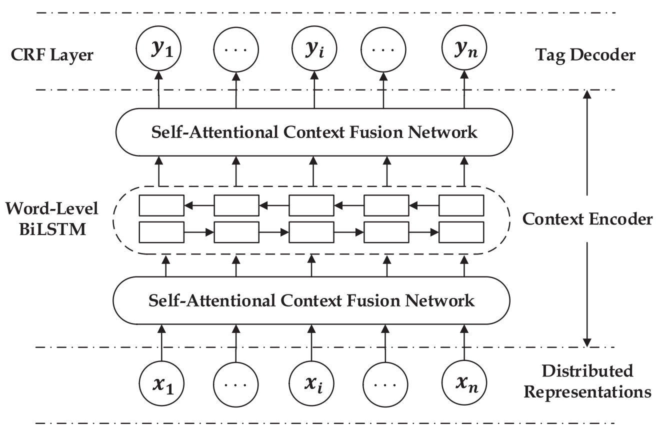


Fig. 2. Overview of proposed neural architecture.

where $\mathbf{Y}(\mathbf{Z})$ is the set of possible label sequences for $\mathbf{Z}$; $\mathbf{W}_{y_{j-1},y_j}$ and b_{y_{j-1},y_j} indicate the weighted matrix and bias parameters corresponding to the label pair (y_{j-1}, y_j), respectively. Then, we employ a likelihood function $\mathcal{L}$ to minimize the negative *log* probability of the golden tag sequence for training,

$$\mathcal{L} = -\sum_{\hat{\mathbf{x}}\in\mathcal{X};\hat{\mathbf{y}}\in\mathcal{Y}} \log p(\hat{\mathbf{y}}|\hat{\mathbf{x}}), \tag{3}$$

where $\mathcal{X}$ denotes the set of training instances, and $\mathcal{Y}$ indicates the corresponding tag set.

Besides pre-trained word embeddings for the basic input, several studies [17,24,33] incorporate character-level representations for exploiting useful intra-word information such as prefix and suffix.

are variants of RNNs designed to cope with gradient vanishing/exploding problems and has become a widely-used context encoder architecture in sequence labeling tasks. *Bi-directional LSTM-CRF* model adopts a LSTM in both directions, generating a forward hidden state and a backward hidden state for each word to extract past and future context information. Then the two hidden states are concatenated to give global information of the whole sequence.

However, we argue that the sequential way to process the inputs of RNNs might weaken the sensitivity of models to context dependencies. Because it recursively compose each word with its forward/backward hidden state that encodes the entire history/future information, and the latent relationship among words is not well extracted to some extent, which is closely related to the final prediction task.

4. Proposed approach

As aforementioned, RNN has limitations in modeling discrete context dependencies of the given sequence, thus in this paper we mainly focus on how to effectively model this kind of context dependencies during the *context encoder* stage within LSTM-CRF architecture (*cf* Section 3). Therefore, we propose a new neural architecture for sequence labeling (shown in Fig. 2), with a novel self-attentional context fusion layer that provides the complementary context information. Specifically, there are two context fusion layers are incorporated at different levels in our proposed architecture, *i.e.,* the one is used for re-weighting the initial input (following the layer of distributed representations), and the other is added for re-weighting the output of *word*-level Bi-LSTM layer. The overall learning process of the proposed self-attentional context fusion network is illustrated in Algorithm 1. Besides, a well-designed *position*-aware self-attention mechanism with three different positional factors is also incorporated into the layer, which models the discrete context dependencies via exploring the relative position information of tokens in a flexible manner.

Algorithm 1 Learning processes of self-attentional context fusion network.

Input: The original token representations of sequence $\mathbf{X} = [\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n]$
Output: The context-aware representations of sequence $\tilde{\mathbf{X}} = [\tilde{\hat{x}}_1, \tilde{\hat{x}}_2, \ldots, \tilde{\hat{x}}_n]$

```
for i ∈ {1...n} do
    for j ∈ {1...n} do
        // Compute the alignment score of x̂_i and x̂_j
        Update f(x̂_i, x̂_j) based on Equation 5
        // Compute the positional bias
        Update Ψ_ij(x̂_i) based on Equation 4
        Update f(x̂_i, x̂_j) by adding Ψ_ij(x̂_i)
    end for
    for j ∈ {1...n} do
        a_i(j) = Softmax(f(x̂_i, x̂_j))
    end for
    ŝ_i ← Sum(a_i(j) ⊙ x̂_j)
    s̃̂_i ← Fullyconnect(ŝ_i)
    // Fusion gate mechanism
    Update λ based on Equation 13
    Update x̃̂_i based on Equation 14
end for
```

Next, we will elaborate our proposed sequence labeling model in detail. More concretely, Section 4.1 will present the proposed *position*-aware self-attention mechanism, followed with the illustration of the proposed context fusion layer in Section 4.2.

4.1. Position-aware self-attention

In this section, we present a novel *position*-aware self-attention for better inducing the importance of each token to a specified token within the same sequence.

Position modeling is benefit for optimizing the self-attention network, since self-attention cannot encode position information of tokens in sequence. Although the position information is implicitly encoded by LSTM in our neural architecture, the process of calculating alignment scores within self-attention is independent of the relative distance of tokens. To this end, here we explore the positional information of an input sequence to extend self-attention model with a different and novel method, aiming to better model the discrete context dependencies of sequence. To be specific, we introduce three different *positional* factors, *i.e., self-disabled mask* bias $\mathbf{M}_{ij}(.)$, *distance-aware Gaussian* bias $\mathbf{G}_{ij}(.)$ and *token-specific position* bias $\mathbf{P}_{ij}(.)$, which are combined in a global positional bias function $\Psi_{ij}(.)$, and added to the baseline self-attention. The three factors are combined by

$$\Psi_{ij}(\hat{\mathbf{x}}_i) = \mathbf{M}_{ij}(\hat{\mathbf{x}}_i) + \alpha\mathbf{G}_{ij}(\hat{\mathbf{x}}_i) + (1-\alpha)\mathbf{P}_{ij}(\hat{\mathbf{x}}_i), \tag{4}$$

where α is a trainable trade-off parameter that controls the contributions of different biases.

The *self-disabled mask* bias $\mathbf{M}_{ij}(.)$ disables the attention of each token to itself, for better measuring its dependency on other tokens. The *distance-aware Gaussian* bias $\mathbf{G}_{ij}(.)$ considers the information of relative distance by utilizing the form of Gaussian distribution, and explicitly affect the computation of attention weights. The *token-specific position* bias $\mathbf{P}_{ij}(.)$ further addresses the interactions between the representations of relative positions and the input presentations, thus explores the relative distance in a more flexible manner. The details of these three factors will be illustrated in the following sections.

More concretely, assume the token representations of sequence $\mathbf{X} = [\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \ldots, \hat{\mathbf{x}}_n]^\top$ with $\hat{\mathbf{x}}_i \in \mathbb{R}^d$. To measure the attention weight of each $\hat{\mathbf{x}}_j$ to a specified token $\hat{\mathbf{x}}_i$, a compatibility function $f(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j)$ is employed to measure the pairwise similarity (*i.e.,* the alignment score) of $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}_j$.

Many different self-attention mechanisms are proposed but are different in the compatibility function $f(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j)$, here we adopt *additive* attention mechanism [2], which is implemented by a one-layer feed-forward neural network and is often superior to others in practice, which is computed by

$$f(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j) = \hat{\mathbf{w}}^\top \sigma(\mathbf{W}^{(1)}\hat{\mathbf{x}}_i + \mathbf{W}^{(2)}\hat{\mathbf{x}}_j + \hat{\mathbf{b}}), \tag{5}$$

where $\sigma(\cdot)$ is an activation function; $\mathbf{W}^{(1)}, \mathbf{W}^{(2)} \in \mathbb{R}^{d\times d}$ indict the weight matrices; $\hat{\mathbf{w}} \in \mathbb{R}^d$ is a weight vector, and $\hat{\mathbf{b}}$ denotes the bias vector.

For effectively encoding position information, we incorporate the proposed positional bias function $\Psi_{ij}(.)$ to $f(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j)$, and the position-aware self-attention is rewritten by

$$f(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j) = w^\top \sigma(\mathbf{W}^{(1)}\hat{\mathbf{x}}_i + \mathbf{W}^{(2)}\hat{\mathbf{x}}_j + \hat{\mathbf{b}}) + \Psi_{ij}(\hat{\mathbf{x}}_i), \tag{6}$$

Then the alignment score is converted by a *softmax* function with the normalization of all the n elements within $\mathbf{X}$, *i.e.,*

$$a_i(j) = \frac{\exp(f(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j))}{\sum_{j'} \exp(f(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_{j'}))}. \tag{7}$$

Finally, the output ($\hat{\mathbf{s}}_i \in \mathbb{R}^d$) of the self-attention of $\hat{\mathbf{x}}_i$ is a weighted sum of representations of all tokens in $\mathbf{X}$ according to the alignment scores, namely,

$$\hat{\mathbf{s}}_i = \sum_{j=1}^{n} a_i(j) \odot \hat{\mathbf{x}}_j. \tag{8}$$

4.1.1. Self-disabled mask bias

For a specific token x_i, the goal of our self-attentional model is to measure its dependency on other tokens in the same sequence and further capture discrete context information, thus it is benefit to prevent the interference of itself information when calculating alignment scores, through disabling the attention of each token to itself. As such, we adopt *self-disabled mask* [35] for self-attention, which is

$$\mathbf{M}_{ij}(\hat{\mathbf{x}}_i) = \begin{cases} 0, & i \neq j, \\ -\infty, & i = j, \end{cases} \tag{9}$$

where $-\infty$ is used to neglect itself contribution in self-attention.

4.1.2. Distance-aware Gaussian bias

Self-attention mechanism models the *global* dependencies among input tokens regardless of their distance, while the relative position information is important for modeling the *local* context in sequence labeling tasks. Without loss of generality, we take *POS tagging* as an example, the POS tag of a word is more likely influenced by its neighbors, as compared with other long-distance words. In order to favor the modeling of short-range dependencies by self-attention, we take account of a *distance-aware Gaussian* bias to control the scope of local context of a specified token x_i, and by incorporating it into the compatibility function, we make the relative distance among tokens to explicitly affect the computation of their corresponding attention weights. The distance-aware Gaussian bias is defined as

$$\mathbf{G}_{ij}(\hat{\mathbf{x}}_i) = \frac{-(i-j)^2}{2\varepsilon^2}, \tag{10}$$

where i, j indicates the order of $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}_j$; parameter ε refers to the standard deviation that is empirically set as $\varepsilon = \frac{k}{2}$; and k is a window size, which is set as 10 in our experiments.

4.1.3. Token-specific position bias

Gaussian bias only takes into account the information of relative distance among tokens, however the way a relative distance affects the distribution of attention might not be the same for different focused tokens, and the discrete context dependencies within the sequence also have much diversity. As such, modeling of the relative distance should be further explored in a more flexible manner for addressing the interactions between the representations of relative positions and the input presentations.

Inspired by Shaw's work [34], we learn a relative position representation matrix $\mathbf{R} \in \mathbb{R}^{r\times d}$ and inject the position information into the attention score. Here d denotes the representation dim and r is a nonnegative value that reflects the maximum margin between two different tokens. In other words, the relative distance between two tokens would be clipped to r if it is greater than the threshold, following the essential hypothesis that the precise relative position information is not useful while beyond a certain distance. And its value is equal to the window size k[3] Specifically, a scalar $\mathbf{P}_{ij}(\hat{\mathbf{x}}_\mathbf{i})$ is composed of two term, which are parameterized as follows,

$$\mathbf{P}_{ij}(\hat{\mathbf{x}}_i) = \begin{cases} \hat{\mathbf{x}}_i^\top \mathbf{R}_r + (\hat{\mathbf{v}}^\top \mathbf{R}_r + \hat{\mathbf{b}}), & |i-j| > r, \\ \hat{\mathbf{x}}_i^\top \mathbf{R}_{|i-j|} + (\hat{\mathbf{v}}^\top \mathbf{R}_{|i-j|} + \hat{\mathbf{b}}), & |i-j| \leq r. \end{cases} \tag{11}$$

where $\hat{\mathbf{v}} \in \mathbb{R}^d$ is a weight vector, and $\hat{\mathbf{b}} \in \mathbb{R}^1$ denotes the bias term. Here the first term is computed by the inner product of $\hat{\mathbf{x}}_i$ and $|i-j|$th (or rth) element of $\mathbf{R}$, which represent the content-dependent positional information. And the second term that transforms the corresponding representation of relative position to a scalar score, can be reparded as a general global position bias. Note that the relative position representation matrix $\mathbf{R}$ is also trainable which is optimized during training along with other parameters.

4.2. Self-attentional context fusion layer

The success of neural networks stems from their highly flexible non-linear transformations. Attention mechanism utilizes a weighted sum to generate the output vectors, which limits its representational ability. To further enhance the power of feature extraction of the attentional layer, we take account of employing two fully connected layers to transform the outputs of the attention module, which is formally computed by

$$\tilde{\mathbf{s}}_i = \tanh[\mathbf{W}^{(\mathbf{z2})} \tanh(\mathbf{W}^{(\mathbf{z1})}\hat{\mathbf{s}}_i + \hat{\mathbf{b}})], \tag{12}$$

where $\mathbf{W}^{(\mathbf{z1})}, \mathbf{W}^{(\mathbf{z2})} \in \mathbb{R}^{d\times d}$ are trainable matrices; and $\hat{\mathbf{s}}_i$ denotes the output of the self-attention of $\hat{\mathbf{x}}_i$ (*cf* Eq. (8)).

As we introduce a *self-disabled mask* (*cf* Section 4.1) to disable the attention of each token to itself, the output of the proposed self-attention layer is insufficient for learning context-aware representation. As such, we propose a feature-wise fusion gate mechanism to adaptively combine the feature of each token with its context. Hence, the final context-aware representation (shown in Fig. 3) of x_i is linearly combine with input of the self-attention layer $\hat{\mathbf{x}}_i$ and the output of the fully connected layers $\tilde{\mathbf{s}}_i$, namely

$$\lambda = \mathrm{sigmoid}(\mathbf{W}^{(\mathbf{f3})} \tanh(\mathbf{W}^{(\mathbf{f1})}\hat{\mathbf{x}}_i + \mathbf{W}^{(\mathbf{f2})}\tilde{\mathbf{s}}_i)) \tag{13}$$

$$\tilde{\mathbf{x}}_i = \lambda \odot \hat{\mathbf{x}}_i + (1-\lambda) \odot \tilde{\mathbf{s}}_i \tag{14}$$

where $\mathbf{W}^{(\mathbf{f1})}, \mathbf{W}^{(\mathbf{f2})}, \mathbf{W}^{(\mathbf{f3})} \in \mathbb{R}^{d\times d}$ are trainable weight matrices of the fusion gate. Note that the learned parameter λ is a vector that has the same dimension with $\tilde{\mathbf{s}}_i$, because different features of $\tilde{\mathbf{s}}_i$ can contain different information of discrete context dependency. Hence the designed fusion gate is able to dynamically select useful information from the self-attention layer in a fine-grained manner.

[3] Note that in the remainder it has the similar meaning when the context is clear and discriminative.

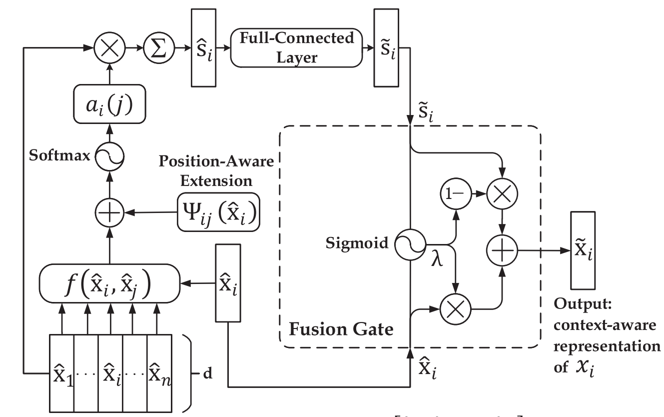


Fig. 3. Self-attentional context fusion network.

Table 1
Statistics of *CoNLL03 NER, WSJ, CoNLL00 chunking* and *OntoNotes 5.0*.

Corpus	Type	Train	Dev	Test
CoNLL03 NER	Sentences	14,987	3,466	3,684
	Tokens	204,567	51,578	46,666
WSJ	Sentences	38,219	5,527	5,462
	Tokens	912,344	131,768	129,654
CoNLL00 chunking	Sentences	8,936	1,000	2,012
	Tokens	211,727	24,294	47,377
OntoNotes 5.0	Sentences	59,924	8,528	8,262
	Tokens	1,088,503	147,724	152,728

5. Experiments

5.1. Data sets

We use four benchmark sequence labeling datasets for evaluation, *i.e., CoNLL 2003 NER dataset* (CoNLL03 NER), *the Wall Street Journal portion of Penn Treebank dataset* (WSJ), *CoNLL 2000 chunking dataset* (CoNLL00 chunking) and *OntoNotes 5.0 English NER datasets* (OntoNotes 5.0). The details about such corpora are shown in Table 1.

- **CoNLL03 NER** is a collection of news wire articles from the *Reuters* corpus, which includes four different types of named entities: PER, LOC, ORG, and MISC. We use the standard dataset split [7] and follow *BIOES* tagging scheme (B, I, O, E, S).
- **WSJ** contains 25 sections and classifies each word into 45 different types of POS tags. Here, we also adopt a standard data split method used in [48], namely, sections 0–18 as *training* data, 19–21 as *development* data, and sections 22–24 as *test* data.
- **CoNLL00 chunking** uses sections 15–18 from the Wall Street Journal corpus for *training* and section 20 for *testing*. It defines 11 syntactic chunk types (*e.g.,* NP, VP, ADJP) in addition to other. Following previous works [29], we randomly sampled 1000 sentences from the training set as *development* data.
- **OntoNotes 5.0** is much larger than *CoNLL 2003 NER dataset*, and consists of text from a wide variety of sources (broadcast conversation, newswire, magazine, Web text, etc.). It is tagged with eighteen entity types (PERSON, ORG, GPE, LAW, etc.). Following previous works [5], we adopt the portion of the dataset with gold-standard named entity annotations, in which the New Testaments portion is excluded.

5.2. Experimental setting

We use LSTM to learn character-level representation of words, and together with the pre-trained word embedding contributes to the distributed representation for input. Then we initialize word embedding with 100-dimensional GloVe [28] and randomly initialize 30-dimensional character embedding. Fine-tuning strategy is adopted that we modify initial word embedding during gradient updates of the neural network model by back-propagating gradients. The size of hidden state of character and word-level Bi-LSTM are set to 100 and 300, respectively. And we fix the number of Bi-LSTM layer as 1 in our neural architecture. All weight matrices in our model are initialized by Glorot Initialization [11], and the bias parameters are initialized with 0.

We train the model parameters by the *mini-batch* stochastic gradient descent (SGD) with momentum. The batch size and the momentum are set to 10 and 0.9, respectively. The leaning rate is updated with $\eta_t = \frac{\eta_0}{1+\rho t}$, where η_0 is the initial learning rate (0.01 for POS tagging and 0.015 for NER and chunking), t is the number of epoch completed and $\rho = 0.05$ is the decay rate. The dropout strategy is used for overcoming the over-fitting on the input and the output of Bi-LSTM with a rate of 0.55, as well as the output of self-attention with a rate of 0.2. We also use gradient clipping [26] to avoid gradient explosion problem. The threshold of gradient norm is set as 5.0. Early stopping [18] is applied for training models according to their performances on development sets.

5.3. Evaluation results and analysis

5.3.1. Over performance

This experiment is to evaluate the effectiveness of sequence labeling on different datasets by our approach. Specifically, we report standard F1-score for CoNLL 2003 NER, CoNLL 2000 chunking and OntoNotes NER tasks, and accuracy for POS tagging task on WSJ. In order to enhance the fairness of the comparisons and verify the solidity of our improvement, we rerun 5 times with different random initialization and report both average and max results of our proposed model as well as our re-implemented Bi-LSTM-CRF baseline. The comparison methods used in this work are the state-of-arts in recent years that usually compared in many previous work. The results for these four tasks are given in Tables 2–5, respectively. Note we do not compare all of models listed in Tables 2 and 4, as such methods (with *) utilize external knowledge excluding in the setting of training set, like *character type* and *lexicon* features [5], shared information learned from other tasks [44], other language models pre-trained from large unlabeled corpus [8].

Specifically, among the models listed in these tables, Collobert et al.[7] employ a simple feed-forward neural network with a fixed-size window for context feature extraction, and adopt CRF method for jointly label decoding; Huang et al.[13] introduce a Bi-LSTM-CRF model and outperform [7] by 0.51% and 0.14% on the dataset of CoNLL03 NER and CoNLL00 chunking, respectively, since Bi-LSTM has good characteristics in modeling sequential data and can better capture contextual information than window-based feed-forward neural network; Lample et al.[17] utilize the same architecture as baseline and further apply a LSTM layer to extract character level features of words, which outperform [13] by 0.84% for CoNLL03 NER task; Similarly, Ma and Hovy [24] achieve a significant improvement of 1.11% over [13] on CoNLL03 NER dataset by equipping the Bi-LSTM-CRF model with a CNN layer to obtain character-level representations of words, which indicates the importance of exploiting useful intra-word information, and their proposed model also becomes a popular baseline for most subsequent work in this field; Zhang et al. [49] propose a method called Multi-Order BiLSTM which combines low order and high order LSTMs together in order to learn more tag dependencies, and

Table 2
Comparison of overall performance on CoNLL 2003 NER task. Note that methods labelled with * indicate that external knowledge are used and thus will not be compared for fairness in our experiments.

Index & Model	F1-score	
	Type	Value(± std[a])
Collobert et al. [7]	reported	89.59
Passos et al. [27]	reported	90.90
Huang et al. [13]	reported	90.10
Lample et al. [17]	reported	90.94
Ma and Hovy [24]	reported	91.21
Rei [32]	reported	86.26
Zhang et al. [49]	reported	90.70
Zhang et al. [48]	reported	91.22
Liu et al. [22][b]	avg	91.24 ± 0.12
	max	91.35
Bi-LSTM-CRF[c] [13]	avg	91.01 ± 0.21
	max	91.27
Our model	avg	91.33 ± 0.08
	max	**91.42**
Chiu and Nichols [5]*	reported	91.62 ± 0.33
Yang et al. [44]*	reported	91.26
Peters et al. [29]*	reported	91.93 ± 0.19
Peters et al. [30]*	reported	92.22 ± 0.10
Devlin et al. [8]*	reported	92.80
Akbik et al. [1]*	reported	93.09 ± 0.12

[a] *std* means Standard Deviation.
[b] Here we do not report the result used in [22], but update it with the result according to the first author's github https://github.com/LiyuanLucasLiu/LM-LSTM-CRF, where the author claimed that the original result is not correct.
[c] Here we re-implement the classical Bi-LSTM model using the same model setting and optimization method with our model.

Table 3
POS tagging accuracy of our model on test data from WSJ proportion of PTB, together with top-performance systems.

Index & Model	Accuracy	
	Type	Value(± std)
Collobert et al. [7]	reported	97.29
Santos and Zadrozny [33]	reported	97.32
Sun [40]	reported	97.36
Søgaard [37]	reported	97.50
Rei [32]	reported	97.43
Ma and Hovy [24]	reported	97.55
Yasunaga et al. [45]	reported	97.58
Liu et al. [22]	avg	97.53 ± 0.03
	max	97.59
Zhang et al. [48]	reported	97.59
Bi-LSTM-CRF [13]	avg	97.51 ± 0.04
	max	97.56
Our model	avg	97.59 ± 0.02
	max	**97.63**

Table 4
Comparison of overall performance on CoNLL00 chunking datatset. Note that methods labelled with * indicate that external knowledge are used and thus will not be compared for fairness in our experiments.

Index & Model	F1-score	
	Type	Value(± std)
Collobert et al. [7]	reported	94.32
Sun et al. [41]	reported	94.52
Huang et al. [13]	reported	94.46
Ma and Sun [23]	reported	94.80
Rei, 2017 [32]	reported	93.88
Zhai et al. [47]	reported	94.72
Zhang et al. [49]	reported	95.01
Bi-LSTM-CRF [13]	avg	94.92 ± 0.08
	max	95.01
Our model	avg	95.09 ± 0.04
	max	**95.15**
Yang et al. [44]*	reported	95.41
Peters et al. [29]*	reported	96.37 ± 0.05
Akbik et al. [1]*	reported	96.72 ± 0.05

Table 5
Comparison of overall performance on OntoNotes 5.0 English NER datasets.

Index & Model	F1-score	
	Type	Value(± std)
Durrett and Klein [9]	reported	84.04
Chiu and Nichols [5]	reported	86.28 ± 0.26
Strubell et al., 2017 [39]	reported	86.84 ± 0.19
Li et al. [20]	reported	87.21
Shen et al. [36]	reported	86.63 ± 0.49
Ghaddar and Langlais [10]	reported	87.95
Bi-LSTM-CRF [13]	avg	87.64 ± 0.23
	max	87.80
Our model	avg	88.12 ± 0.22
	max	**88.33**

Table 6
Experimental results of various position modeling strategies applied to self-attention.

No	Model	F1-score ± std
1	w/o $\Psi_{ij}(\hat{\mathbf{x}}_i)$ in Eq. (6)	91.15 ± 0.12
2	add position encoding	91.05 ± 0.19
3	Our model	91.33 ± 0.08

this method outperforms [13] by 0.6% and 0.55% on the dataset of CoNLL03 NER and CoNLL00 chunking, however, it yields a worse performance than [24]; Zhang et al. [48] propose a multi-channel model that performs better than [24] with a slight improvement of 0.01% and 0.04% on CoNLL03 NER and WSJ dataset, which takes the long range tag dependencies into consideration by incorporating a tag LSTM in their model; Liu et al. [22] incorporate character-aware neural language models into the Bi-LSTM-CRF model and outperform [24] by 0.02% on CoNLL03 NER task, but fail to achieve a better performance for POS tagging.

Note the results show that our proposed model outperforms Bi-LSTM-CRF model by 0.32%, 0.08%, 0.17% and 0.48% for the dataset of CoNLL03 NER, WSJ POS tagging, CoNLL00 chunking and OntoNotes 5.0, respectively, which could be viewed as significant improvements in the filed of sequence labeling. Even compared with the top-performance popular baseline [24], our model achieves a much better result for both NER and POS tagging tasks than other top-conference work in recent two years [48], with an improvement of 0.12% and 0.04%, respectively. Besides, the std (Standard Deviation) value of our model is smaller than the one of Bi-LSTM-CRF, which demonstrates our proposed method is more robust. We also observe that our model consistently outperforms all these baselines for different tasks. Because such models mostly adopt Bi-LSTM as their context encoder architecture, which cannot directly induce the relations among two words, and thus omit modeling part of context dependency especially some discrete patterns. By proposing a novel *position*-aware self-attention and incorporating self-attentional context fusion layers into the neural architecture, our proposed model is capable of extracting the sufficient latent relationship among words, thus can provide the complementary context information on the basis of Bi-LSTM.

5.3.2. Ablation study

In this section, we run experiments on the CoNLL 2003 NER dataset to dissect the relative impact of each modeling decision by ablation studies.

For better understanding the effectiveness of our proposed *position*-aware self-attention in our model, we evaluate the performance of various position modeling strategies. Training process is performed 5 times, and then the average F1-scores are reported in Table 6. Note that Model 3 is our final proposed architecture. Model 1 remains the same as Model 3 except that it minus $\Psi_{ij}(\hat{\mathbf{x}}_i)$ in Eq. (6), which suggests there exists no position information

Table 7
Experimental results for ablating three positional factors.

$\mathbf{M}_{ij}(\hat{\mathbf{x}}_i)$	$\mathbf{P}_{ij}(\hat{\mathbf{x}}_i)$	$\mathbf{G}_{ij}(\hat{\mathbf{x}}_i)$	F1-score ± std
×	✓	✓	91.12 ± 0.21
✓	×	✓	91.07 ± 0.05
✓	✓	×	91.19 ± 0.24
✓	✓	✓	91.33 ± 0.08

Table 8
Experimental results for ablating two self-attentional context fusion layer.

First layer	Second layer	F1-score ± std
×	×	91.01 ± 0.21
×	✓	91.13 ± 0.17
✓	×	91.27 ± 0.05
✓	✓	91.33 ± 0.08

within self-attention. Model 2 applies an absolute position encoding before context encoder layer on the basis of Model 1, which is the position modeling strategy adopted by Vaswani et al. [43] in the Transformer model. Comparing Model 1 with Model 3, we can see that after removing the proposed positional bias $\Psi_{ij}(\hat{\mathbf{x}}_i)$ the performance decreases a lot, indicating that our proposed flexible extension of the self-attention achieves a significant improvement since it effectively explores the positional information of an input sequence. But Model 2 with absolute position encoding yields worse performance than Model 1. We conjecture that it is because the absolute position embedding might weaken model's ability to fusion context features in our architecture.

In order to better understand the working mechanism of our proposed *position*-aware self-attention, we further analysis the influence of three different positional factors incorporated in it. One of the three factors is removed from proposed positional bias function (Eq. (4)) each time and the results are shown in Table 7. We can clearly see that the final proposed model including all three factors achieves the best performance and ablating any one bias contributes to a worse score. It demonstrates the effectiveness of our well designed positional bias to explore the relative position information of tokens from different perspectives. The result also shows that after removing $\mathbf{P}_{ij}(\hat{\mathbf{x}}_i)$, the F1-score decreases the most, indicating the *token-specific position* bias leads to a significantly better performance since it considers the relative positions in a more flexible manner by tacking the interactions with input representations and has advantages in modeling discrete context dependencies.

In addition, in order to investigate the influence of our designed self-attentional context fusion layer, we also conduct ablation tests where one of the two layers (*cf* Fig. 2) is removed from our neural architecture each time. Table 8 shows that including either one self-attentional context fusion layer contributes to an obvious improvement over the baseline model, which verifies the effectiveness of our proposed self-attentional context fusion layer to provide the complementary context information at different levels and then enhance the prediction.

As can be seen from Fig. 2, the context encoder of proposed neural architecture consists of three parts, *i.e.*, bottom, middle and top. And we conduct corresponding experiment to illustrate why Bi-LSTM is placed between two self-attentional layers. The results are given in Table 9. Note that Model 4 is our final proposed architecture, and comparing Model 1 with Model 4, we can see that after removing the Bi-LSTM layer the performance decreases a lot, indicating the powerful ability of Bi-LSTM to capture sequential long-term dependencies. By comparing Model 2 and Model 3 with Model 4, it can be found that replacing either self-attentional layer in the architecture with Bi-LSTM will not lead to an improvement in results, further illustrating the effectiveness and rationality of the designed architecture. Finally, we also shows the necessity of adopting CRF instead of softmax as the decoder by comparing Model 5 and Model 4.

Table 9
Experimental results for adjusting the architecture of the proposed model (SAN denotes the proposed self-attentional context fusion network).

No	Context Encoder			Decoder	F1-score ± std
	Bottom	Middle	Top		
1	SAN	SAN	SAN	CRF	90.25 ± 0.19
2	Bi-LSTM	Bi-LSTM	SAN	CRF	91.06 ± 0.10
3	SAN	Bi-LSTM	Bi-LSTM	CRF	91.19 ± 0.12
4	SAN	Bi-LSTM	SAN	CRF	91.33 ± 0.08
5	SAN	Bi-LSTM	SAN	Softmax	88.79 ± 0.26

Table 10
Experimental results of the Transformer model.

Num of layers	Position modeling strategy	F1-score ± std
1	Absolute position embedding	88.7 ± 0.23
1	Proposed positional bias	90.69 ± 0.21
2	Absolute position embedding	88.79 ± 0.39
2	Proposed positional bias	90.7 ± 0.15
3	Absolute position embedding	88.57 ± 0.23
3	Proposed positional bias	90.6 ± 0.12

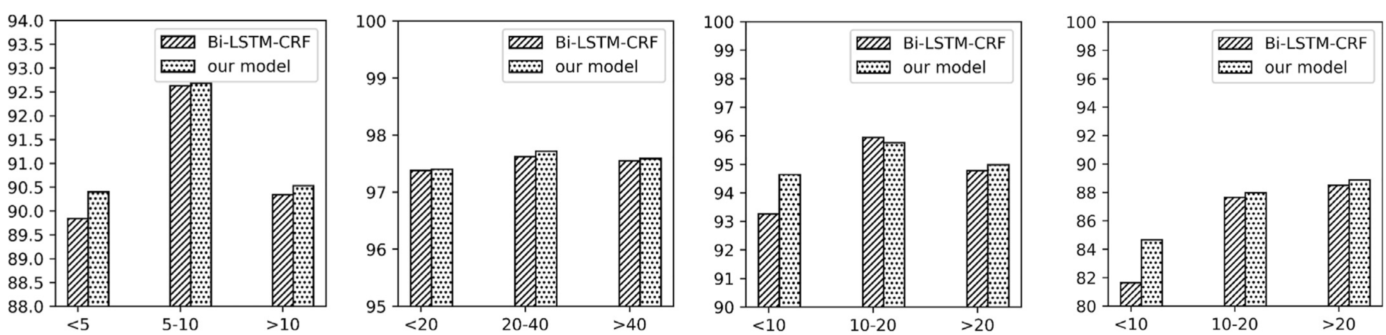
The Transformer model [43] which is based on self-attention mechanism has been proven to have strong capabilities for feature extraction. We evaluate the transformer with different numbers of layers on CoNLL03 NER task, and the result is given in Table 10. In the experiment we adopt the transformer as the context encoder architecture and remain the distributed representations and tag decoder part of our model. And we also evaluate the performance of various position modeling strategies on the Transformer architecture, in which the proposed positional bias is used to replace the absolute position encoding. Table 10 shows that all these models yield a poor performance that even worse than most of our baselines. We conjecture that it's because the transformer model may be sensitive to the hyper-parameters for different sequence labeling tasks, since there are lots of hyper-parameters like dimension of keys/queries/values, dimension of attention model, dimension of inner-layer, number of heads and etc. As for the setting of this experiment, the parameters are set to 64, 512, 1024 and 8, respectively. However, it's obvious that changing the position modeling strategy leads to an greate improvement to the results, which further demonstrate the effectiveness of our proposed method to explore position information for sequence labeling tasks.

5.3.3. Performances on different length

We further analyze the performance of different models with respect to the different length of sentences. In Fig. 4, we compare Bi-LSTM-CRF baseline and our proposed model on different sentence lengths. For NER and chunking, our model significantly outperforms Bi-LSTM-CRF on short sentences (sentence length less than 5 on CoNLL2003, length less than 10 on CoNLL2000 and OntoNotes), which indicates that the improvement of the proposed model on short sentences is much larger than those on long sentences. The discrete context dependencies with short distances in a sequence are captured very well by our proposed model but simply neglected by Bi-LSTM-CRF. For POS tagging, performances of the two models on different sentence lengths are relatively comparable, while in the range of (20;40) our model performs slightly better than Bi-LSTM-CRF.

Table 11
Examples of the predictions of Bi-LSTM-CRF baseline and our model.

Sent1	The market opened sharply **lower**, with the Nikkei average down nearly 600 after 20 min																
Gold	DT	NN	VBD	RB	[RBR]	IN	DT	NNP	NN	RB	RB	CD	IN	CD	NNS		
Bi-LSTM	DT	NN	VBD	RB	JJR	IN	DT	NNP	NN	RB	RB	CD	IN	CD	NNS		
Our Model	DT	NN	VBD	RB	[RBR]	IN	DT	NNP	NN	RB	RB	CD	N	CD	NNS		
Sent2	The dollar also moved **higher** in Tokyo.																
Gold	DT	NN	RB	VBD	[RBR]	IN	NNP										
Bi-LSTM	DT	NN	RB	VBD	JJR	IN	NNP										
Our Model	DT	NN	RB	VBD	[RBR]	IN	NNP										
Sent3	But the rally was confined to the stocks, which had been hard **hit** during Friday 's selling frenzy.																
Gold	CC	DT	NN	VBD	VBN	TO DT	NNS	WDT	VBD	VBN	RB [VBN]	IN	NNP	POS	NN		NN
Bi-LSTM	CC	DT	NN	VBD	VBN	TO DT	NNS	WDT VBD VBN	RB	NN	IN	NNP	POS NN	NN			
Our Model	CC	DT	NN	VBD	VBN	TO DT	NNS	WDT	VBD	VBN	RB	[VBN]	IN	NNP	POS	NN	NN
Sent4	Spending patterns in newspapers have been **upset** by shifts in ownership and general hardships.																
Gold	NN	NNS	IN	NNS	VBP	VBN	[VBN]	IN	NNS	IN	NN	CC	JJ	NNS			
Bi-LSTM	NN	NNS	IN	NNS	VBPVBN			JJ	IN	NNS	IN	NN	CC	JJ	NNS		
Our Model	NN	NNS	IN	NNS	VBP	BN	[VBN]	IN	NNS	IN	NN	CC	JJ	NNS			



(a) CoNLL03 NER (b) WSJ POS tagging (c) CoNL00 Chunking (d) OntoNotes 5.0

Fig. 4. Performances on different lengths.

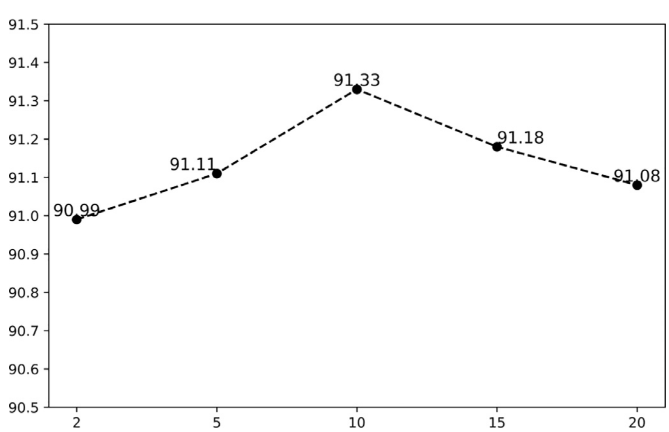


Fig. 5. Performance of our model with various window sizes.

5.3.4. Impact of window size

The window size k (*cf* Section 4.1.2) is clearly a hyperparameter which must be optimized for, thus we investigate the influence of the value of k on the CoNLL2003 NER task. We also rerun 5 times with different random initialization and report the average score, which is consistent with our other experiments in this paper. The plot in Fig. 5 shows that when assigning the value of k to 10 we do outperform other models substantially. And with other window sizes (except 2) our model performs relatively well and is superior to the Bi-LSTM-CRF baseline (91.01%), which also suggests the effectiveness of our proposed *distance-aware Gaussian* bias to favor the local context dependencies of sequence.

Table 12
Training speed, training time and perfromance of Bi-LSTM-CRF baseline and our proposed model on CoNLL 2003 NER task. N iter/s means processing N iterations per second.

Model	F1-score ± std	speed	ime
Bi-LSTM-CRF	91.01 ± 0.21	23 iter/s	1.6 h
Out model	91.33 ± 0.08	20 iter/s	1.9 h

5.3.5. Efficiency

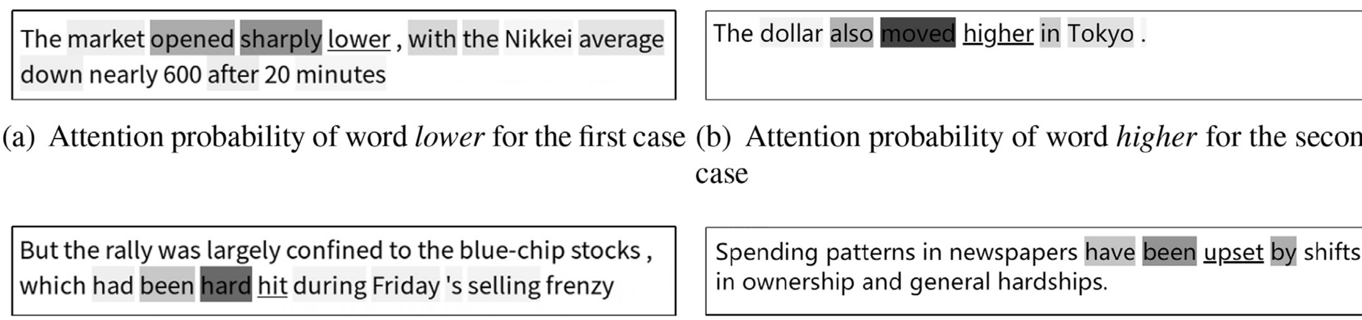
We implement our model based on the PyTorch library. Models have been trained on one GeForce GTX 1080 GPU, with training time recorded in Table 12. In terms of efficiency, our model only introduces a small number of parameters in two self-attention layer, which may not have a very large impact on efficiency. And it can be drawn from Table 12 that the training speed of our model is only 13% lower than the baseline, but bring a significant improvement in the performance.

5.3.6. Qualitative analysis

The weight λ in fusion gate mechanism actually indicates the balance between the feature of each token and its context representation obtained by self-attention. If λ is bigger than 0.5, the final contextual representation relies more on its own feature, otherwise, the context representation play a more important role. As shown in Table 13, λ varies along the sentence, showing the effectiveness of both feature. Besides, we can observe that for most tokens, λ_1 is smaller than λ_2, which indicates that the first self-attentional layer before Bi-LSTM incorporates more useful contextual information.

Table 13
Qualitative analysis of learned parameter λ (*cf* Eq. (14)). λ_1 and λ_2 denote λ in the first and second self-attentional context fusion layer, respectively. Since λ is a multi-dimensional vector, here we take its average in various dimensions to facilitate the observation.

Sent1	Results of Asian Cup group C matches played on Friday										
Tag	O	O	B-MISC	E-MISC O	O	O	O	O	O		
λ_1	0.89	0.8	0.61	0.88	0.75	0.86	0.87	0.87	0.85	0.81	
λ_2	0.93	0.93	0.92	0.86	0.87	0.94	0.89	0.92	0.94	0.94	
Sent2	Japan	-	Hassan	Abbas	84	own	goal	,	Takuya	Takagi	88
Tag	S-LOC	O	B-PER	E-PER	O	O	O	O	B-PER	E-PER	O
λ_1	0.81	0.78	0.73	0.7	0.89	0.83	0.89	0.77	0.74	0.72	0.89
λ_2	0.93	0.87	0.91	0.78	0.83	0.93	0.93	0.93	0.9	0.77	0.8
Sent3	2.	Candice	Gilg	(France)	24.31						
Tag	O	B-PER	E-PER	O	S-LOC	O	O				
λ_1	0.77	0.74	0.74	0.75 0.79 0.73 0.76							
λ_2	0.89	0.82	0.65	0.56 0.77 0.71 0.87							



(a) Attention probability of word *lower* for the first case (b) Attention probability of word *higher* for the second case

(c) Attention probability of word *hit* for the third case (d) Attention probability of word *upset* for the fourth case

Fig. 6. Heatmaps of four cases.

5.3.7. Case study

In this section, we present an in-depth analysis of results given by our proposed approach for better understanding the influence of self-attention mechanism in our proposed model. Without loss of generality, we take *POS tagging* as the task and Bi-LSTM-CRF as the comparison method for comparison. Table 11 shows four cases that our model predicts correctly but Bi-LSTM-CRF doesn't. For better comparison, we visualize the alignment score by *heatmaps* of words that baseline model fails to predict their labels correctly.

In the first case, the POS tag of "lower" should be tagged with *adverb comparative* (RBR), while Bi-LSTM-CRF recognizes it as *adjective comparative* (JJR). It's obvious that the tag of "lower" is dependent on the 3rd word "open", where an adverb is associated with a verb, and the 4th word "sharply" is a direct modifier of it. Fig. 6(a) shows that for word "lower" it pays more attention on "opened" and "sharply", while less on other words. Similar situation is shown in the second case, where our model assigns correct POS tag to "higher" which depends largely on its previous word "moved" but Bi-LSTM-CRF fails. Regarding the third case, our model succeeds in assigning *verb past participle* (VBN) to word "hit" by considering "been" and "hard" while Bi-LSTM-CRF makes a wrong decision. The consistent conclusion is also reflected in Fig. 6(c), that "been" and "hard" obtain large attention from the focus word "hit". And our model predict the POS tag of "upset" correctly in the fourth case which can be speculated from the common phrase "have been done by".

Our analysis suggests if the choice of assigning label to a specified token x_i depends on several other words, they will receive a large amount of attention scores from x_i, which also provides a high level interpretability for our self-attentional model.

6. Conclusions

This paper proposes a innovative neural architecture for sequence labeling tasks, in which a self-attentional context fusion layer is designed and incorporated to better model discrete and discontinuous context patterns of sequence. The strengths of our work are that we identify the problem of modeling discrete context dependencies in sequence labeling tasks, and a position-aware self-attention is proposed to induce the latent independent relations among tokens over the input sequence via three different bias, which can effectively model the context dependencies of given sequence according to the relative distance among tokens. Experimental results on *part-of-speech* (**POS**) *tagging, named entity recognition* (**NER**) and *phrase chunking* tasks demonstrate the effectiveness of our proposed model which achieves *state-of-the-art* performance. Furthermore, our analysis reveals the effects of each modeling decision from different perspectives. The way we model the discrete context dependencies of sentences in sequence labeling tasks can also inspire other researchers in the field to innovate from this perspective. Despite the good performance, our work still has weaknesses, which is reflected in the limited improvement of our model for longer sequences. The main reason is that the second positional bias that we introduced tends to let self-attention learn the influence of neighboring words in the sequence. In the future, we plan to further apply our neural architecture to data from other domains such as social media and empower more sequence labeling tasks. Additionally, we also plan to employ our model to other sequence learning tasks besides sequence labeling, such as event extraction and neural machine translation. More recently, pretrained language models from huge corpus are widely adopted to enhance the representation of words. We will in the future explore integrating language modeling into this architecture to further boosting performance.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.patcog.2020.107636.

References

[1] A. Akbik, D. Blythe, R. Vollgraf, Contextual string embeddings for sequence labeling, in: COLING, 2018, pp. 1638–1649.
[2] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, ICLR, 2014.
[3] L.E. Baum, T. Petrie, Statistical inference for probabilistic functions of finite state Markov chains, Ann. Math. Stat. (1966) 1554–1563.
[4] J. Cheng, D. Li, M. Lapata, Long short-term memory-networks for machine reading, EMNLP, 2016.
[5] J.P. Chiu, E. Nichols, Named entity recognition with bidirectional LSTM-CNNs, TACL 4 (2016) 357–370.
[6] M. Collins, Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms, in: Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing-Volume 10, Association for Computational Linguistics, 2002, pp. 1–8.
[7] R. Collobert, K. Kavukcuoglu, J. Weston, L. Bottou, P. Kuksa, M. Karlen, Natural language processing (almost) from scratch, J. Mach. Learn. Res. 12 (1) (2011) 2493–2537.
[8] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: arXiv, 2018.
[9] G. Durrett, D. Klein, A joint model for entity analysis: coreference, typing, and linking, Trans. Assoc. Comput.Ling. 2 (2014) 477–490.
[10] A. Ghaddar, P. Langlais, Robust lexical features for improved neural network named-entity recognition, COLING, 2018.
[11] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, 2010, pp. 249–256.
[12] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[13] Z. Huang, X. Wei, Y. Kai, Bidirectional LSTM-CRF models for sequence tagging, Computer Science, 2015.
[14] T. Kudoh, Y. Matsumoto, Use of support vector learning for chunk identification, in: Fourth Conference on Computational Natural Language Learning and the Second Learning Language in Logic Workshop, 2000.
[15] S. Kwon, Y. Ko, J. Seo, Effective vector representation for the korean named-entity recognition, Pattern Recognit. Lett. 117 (2019) 52–57.
[16] J. Lafferty, A. McCallum, F.C. Pereira, Conditional random fields: Probabilistic models for segmenting and labeling sequence data, ICML, 2001.
[17] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, C. Dyer, Neural architectures for named entity recognition, NAACL, 2016.
[18] C. Lee, G. Overfitting, R. Caruana, S. Lawrence, L. Giles, Overfitting in neural nets: backpropagation, conjugate gradient, and early stopping, NIPS, 2000.
[19] H.-g. Lee, G. Park, H. Kim, Effective integration of morphological analysis and named entity recognition based on a recurrent neural network, Pattern Recognit. Lett. 112 (2018) 361–365.
[20] P.-H. Li, R.-P. Dong, Y.-S. Wang, J.-C. Chou, W.-Y. 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, 2017, pp. 2664–2669.
[21] Z. Lin, M. Feng, C.N.d. Santos, M. Yu, B. Xiang, B. Zhou, Y. Bengio, A structured self-attentive sentence embedding, ICLR, 2017.
[22] L. Liu, J. Shang, X. Ren, F.F. Xu, H. Gui, J. Peng, J. Han, Empower sequence labeling with task-aware neural language model, AAAI, 2018.
[23] S. Ma, X. Sun, A new recurrent neural CRF for learning non-linear edge features, in: arXiv:1611.04233, 2016.
[24] X. Ma, E. Hovy, End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF, ACL, 2016.
[25] A. Mccallum, W. Li, Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons, CoNLL, 2003.
[26] R. Pascanu, T. Mikolov, Y. Bengio, On the difficulty of training recurrent neural networks, ICML, 2013.
[27] A. Passos, V. Kumar, A. Mccallum, Lexicon infused phrase embeddings for named entity resolution, CS, 2014.
[28] J. Pennington, R. Socher, C. Manning, Glove: Global vectors for word representation, in: EMNLP, 2014, pp. 1532–1543.
[29] M.E. Peters, W. Ammar, C. Bhagavatula, R. Power, Semi-supervised sequence tagging with bidirectional language models, ACL, 2017.
[30] M.E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, L. Zettlemoyer, Deep contextualized word representations, NAACL, 2018.
[31] L.R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE 77 (2) (1989) 257–286.
[32] M. Rei, Semi-supervised multitask learning for sequence labeling, ACL, 2017.
[33] C.N.D. Santos, B. Zadrozny, Learning character-level representations for part-of-speech tagging, ICML, 2014.
[34] P. Shaw, J. Uszkoreit, A. Vaswani, Self-attention with relative position representations, NAACL, 2018.
[35] T. Shen, T. Zhou, G. Long, J. Jiang, S. Pan, C. Zhang, DiSAN: Directional self-attention network for RNN/CNN-free language understanding, AAAI, 2018.
[36] Y. Shen, H. Yun, Z.C. Lipton, Y. Kronrod, A. Anandkumar, Deep active learning for named entity recognition, ICLR, 2018.
[37] A. Søgaard, Semisupervised condensed nearest neighbor for part-of-speech tagging, ACL, 2011.
[38] M. Sperber, J. Niehues, G. Neubig, S. Stüker, A. Waibel, Self-attentional acoustic models, Interspeech, 2018.
[39] E. Strubell, P. Verga, D. Belanger, A. McCallum, Fast and accurate entity recognition with iterated dilated convolutions, EMNLP, 2017.
[40] X. Sun, Structure regularization for structured prediction, NIPS, 2014.
[41] X. Sun, W. Li, H. Wang, Q. Lu, Feature-frequency–adaptive on-line training for fast and accurate natural language processing, Comput. Ling. 40 (3) (2014) 563–586.
[42] Z. Tan, M. Wang, J. Xie, Y. Chen, X. Shi, Deep semantic role labeling with self--attention, AAAI, 2018.
[43] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: NIPS, 2017, pp. 5998–6008.
[44] Z. Yang, R. Salakhutdinov, W.W. Cohen, Transfer learning for sequence tagging with hierarchical recurrent networks, ICLR, 2017.
[45] M. Yasunaga, J. Kasai, D. Radev, Robust multilingual part-of-speech tagging via adversarial training, NAACL, 2017.
[46] S. Yousfi, S.-A. Berrani, C. Garcia, Contribution of recurrent connectionist language models in improving LSTM-based arabic text recognition in videos, Pattern Recognit. 64 (2017) 245–254.
[47] F. Zhai, S. Potdar, B. Xiang, B. Zhou, Neural models for sequence chunking, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017.
[48] Y. Zhang, H. Chen, Y. Zhao, Q. Liu, D. Yin, Learning tag dependencies for sequence tagging., IJCAI, 2018.
[49] Y. Zhang, X. Sun, S. Ma, Y. Yang, X. Ren, Does higher order LSTM have better accuracy for segmenting and labeling sequence data? COLING, 2017.

Wei Wei received the Ph.D. degree from Huazhong University of Science and Technology, Wuhan, China, in 2012. He is currently an associate professor with the School of Computer Science and Technology, Huazhong University of Science and Technology. He was a research fellow with Nanyang Technological University, Singapore, and Singapore Management University, Singapore. His current research interests include artificial intelligence, natural language processing, information retrieval, data mining, machine learning, and social computing and recommender system.

Zanbo Wang received the bachelor degree from Zhongnan University of Economics and Law, Wuhan, China, in 2019. She is currently a master at Huazhong University of Science and Technology, China. Her research interests include natural language processing and deep learning.

Xian-Ling Mao received the Ph.D. degree from Peking University, in 2013. He is currently an associate professor of computer science with the Beijing Institute of Technology. He works in the fields of machine learning and information retrieval. His current research interests include topic modeling, learning to hashing, and question answering. Dr. Mao is a member of the IEEE Computer Society and a member of the Association for ComputingMachinery (ACM).

Guangyou Zhou received the Ph.D. degree from National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences (IACAS) in 2013. Currently, he worked as a Professor at the School of Computer, Central China Normal University. His research interests include natural language processing and information retrieval. He has published more than 30 papers in the related fields.

Pan Zhou received the Ph.D. degree from the Georgia Institute of Technology, America, in 2011. He is currently an associate professor and PhD advisor with Hubei Engineering Research Center on Big Data Security, School of Cyber Science and Engineering, Huazhong University of Science and Technology. His current research interests include big data analysis, data and network security and privacy, and wireless scheduling algorithms. Dr. Zhou is a member of the IEEE Computer Society.

Sheng Jiang received the Ph.D. degree from Huazhong University of Science and Technology, Wuhan, China, in 2009. He is currently a lecture with the School of Computer Science and Technology, Huazhong University of Science and Technology. His current research interests include data mining and natural language processing.