Knowledge-Based Systems 212 (2021) 106548

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

ASRNN: A recurrent neural network with an attention model for sequence labeling

Jerry Chun-Wei Lin [a,b,*], Yinan Shao [c], Youcef Djenouri [d], Unil Yun [e]

[a] School of Information and Control Engineering, Qingdao University of Technology, Qingdao, China
[b] Department of Computer Science, Electrical Engineering and Mathematical Sciences, Western Norway University of Applied Sciences, Bergen, Norway
[c] Alibaba inc., Hangzhou, Zhejiang, China
[d] SINTEF Digital, Oslo, Norway
[e] Department of Computer Engineering, Sejong University, Seoul, Republic of Korea

ARTICLE INFO

Article history:
Received 15 December 2019
Received in revised form 7 July 2020
Accepted 19 October 2020
Available online 6 November 2020

Keywords:
Semi-CRF
Attention mechanism
Sequence labeling
Neural network

ABSTRACT

Natural language processing (NLP) is useful for handling text and speech, and sequence labeling plays an important role by automatically analyzing a sequence (text) to assign category labels to each part. However, the performance of these conventional models depends greatly on hand-crafted features and task-specific knowledge, which is a time consuming task. Several conditional random fields (CRF)-based models for sequence labeling have been presented, but the major limitation is how to use neural networks for extracting useful representations for each unit or segment in the input sequence. In this paper, we propose an attention segmental recurrent neural network (ASRNN) that relies on a hierarchical attention neural semi-Markov conditional random fields (semi-CRF) model for the task of sequence labeling. Our model uses a hierarchical structure to incorporate character-level and word-level information and applies an attention mechanism to both levels. This enables our method to differentiate more important information from less important information when constructing the segmental representation. We evaluated our model on three sequence labeling tasks, including named entity recognition (NER), chunking, and reference parsing. Experimental results show that the proposed model benefited from the hierarchical structure, and it achieved competitive and robust performance on all three sequence labeling tasks.

1. Introduction

Natural language processing (NLP) is useful for handling text and speech. Within NLP, sequence labeling is the important task of identifying and assigning category labels to each unit or subsequence within a given input. Due to its role in several downstream tasks, including relation extraction [1,2], entity linking [3], and coreference resolution [4], it has received substantial attention for several decades. Some conventional sequence labeling models, including the conditional random fields (CRF) [5] and maximum entropy models (MEM) [6], establish the conditional probability over the input sequence by analyzing individual input units (i.e., characters or words). Other sequence labeling approaches, known as segmentation models, including semi-Markov conditional random fields (semi-CRF) model [7], analyze the input at the segment (i.e., subsequence) level. Compared with sequence labeling models, segmentation models capture more segment-level features (i.e., segment length, boundary words, etc.) without limitations from local label dependencies. However, the performance of these conventional models depends greatly on hand-crafted features and task-specific knowledge. In practice, it is time consuming to develop such systems, and the performance of the developed system often declines when it is applied to a new domain or task. Therefore, other researchers have proposed neural network conditional random fields (CRF) based models for sequence labeling [8–11]. Instead of conventional hand-crafted binary features, these neural network CRF methods provide continuous features, require no feature engineering, and offer stable performance for a variety of problems. Compared with conventional CRF models, neural networks are capable of modeling long term dependency context information in the sequence and learning distributed representations from unlabeled training data. A key problem in neural CRF models is how to use neural networks to extract useful representations for each unit or segment in the input sequence.

* Corresponding author at: Department of Computer Science, Electrical Engineering and Mathematical Sciences, Western Norway University of Applied Sciences, Bergen, Norway.
E-mail addresses: jerrylin@ieee.org (J.C.-W. Lin), shaoyn0817@gmail.com (Y. Shao), Youcef.Djenouri@sintef.no (Y. Djenouri), yunei@sejong.ac.kr (U. Yun).

https://doi.org/10.1016/j.knosys.2020.106548
0950-7051/

In this paper, we propose a hierarchical attention neural semi-CRF model for sequence labeling. The proposed model uses a neural-based semi-CRF structure to complete varied tasks of sequence labeling without hand-crafted features or task-specific knowledge. It can not only extract neural features automatically but also incorporate hand-crafted sparse CRF/semi-CRF features easily. In addition, the proposed model balances these two types of features by tuning a learnable parameter in training phrase. The proposed model also incorporates an attention mechanism to attend differently on the input characters/words, which makes the model automatically find the most intuitive characters/words from the input sequences. To find the features at different levels, the proposed model has three layers: a character-level encoder, a word-level encoder, and a semi-CRF layer, from the bottom to the top. The first two encoders are used to extract corresponding character-level and word-level features with the attention mechanism. The upper semi-CRF layer is used to incorporate features at different levels into one graphical model and to jointly model the probability distribution of different sequence labels. An attention segmental recurrent neural network (ASRNN) is used in the lowest character-level encoder layer to extract character-level representation. The output of the character-level encoder layer combines with pre-trained word embeddings and enters the middle word-level encoder layer to extract segment representations. The middle encoder uses ASRNN, which is capable of attending differentially to each word when constructing segmental representations by using the attention mechanism. The extracted segment representations pass to the top semi-Markov conditional random fields (semi-CRF) layer to jointly decode the best label sequence. Our model requires no feature engineering or data processing, making it applicable to a wide variety of sequence labeling tasks. We conducted extensive experiments on three sequence labeling tasks: entity recognition (NER), chunking, and reference parsing. Experimental results show that our model benefits from the hierarchical structure and offers robust performance over a variety of sequence labeling tasks. Our main contributions can be summarized as follows:

- We propose an ASRNN that automatically constructs segment representations by using an attention mechanism to attend differently on individual characters and words.
- We introduce an end-to-end hierarchical attention neural semi-CRF model, which can effectively extract word-level and segment level neural network features. Moreover, to further enhance the model performance, sparse CRF features and semi-CRF features can be easily utilized in the proposed models. We also utilize learnable weights in an objective function to automatically balance neural features and sparse features.
- The designed ASRNN model needs no data processing or task-specific feature engineering, and it shows competitive and robust performance for several different sequence labeling tasks.

2. Literature review

2.1. HMM-based model

The hidden Markov model (HMM) and its extensions were proposed by Baum et al. [12–16], which can be used for representation as a dynamic Bayesian network. When applying an HMM to sequence labeling, the states (i.e., labels) are invisible; the outputs (i.e., words or segments), which are dependent on the states, are visible. Each state has a probability distribution over the possible output tokens, called emission probabilities. Each state also has a probability distribution over possible states, called transition probabilities. The sequence of tokens generated by an HMM model gives the probability distribution over all the possible sequences of states. A forward–backward algorithm is used to find the unknown parameters of an HMM, with the Viterbi algorithm used to find the most likely sequence of hidden states.

Many proposals for sequence labeling have used a HMM. In 1998, Fine et al. [17] proposed a hierarchical hidden Markov model (HHMM), which is a recursive hierarchical generalization of the plain HMM. They used a systematic unsupervised approach to the modeling of the complex multi-scale structure that appears in many natural language texts and derived an efficient procedure for estimating the model parameters from unlabeled data. In 2003, Zhang et al. [18] built the ICTCLAS system using an HHMM to incorporate Chinese word segmentation, part-of-speech tagging, disambiguation, and unknown words recognition into a single theoretical framework. Furthermore, in 2003, Shen et al. [19] proposed a named entity recognizer for biomedical applications using an HMM. Compared with previous works, it provided more useful features, including simple deterministic functions, morphological analysis, POS tagging, and semantic triggers, to better identify the biomedical named entity. They also proposed a simple algorithm to solve the abbreviation problem and a rule-based method to deal with cascaded phenomena in the biomedical domain.

2.2. MEM-based model

The maximum entropy model (MEM) in NLP was introduced in the pioneering work of Berger et al. [6]. They presented a maximum-likelihood approach for automatically constructing maximum-entropy models and estimating the parameters of such models. Many researchers have used MEM for sequence labeling. In 2004, Lim et al. [20] proposed a semantic role labeling method using a maximum entropy model that enabled not only the full use of rich features but also alleviated the data sparseness problem in a well-designed model. Sun et al. [21] proposed a system to solve the joint learning of syntactic and semantic dependencies in 2008. Their proposal used a directed graphical model to integrate dependency relation classification and semantic role labeling. In 2009, Yu et al. [22] investigated the problem of using continuous features in the MEM. They explained why the MEM with the moment constraint (MEMC) works well with binary features but not with continuous features and showed that the weights associated with the continuous features should be continuous functions instead of single values.

In 1996, Ratnaparkhi [23] proposed a statistical model trained from a corpus annotated with part-of-speech tags and assigned them to previously unseen text with high accuracy. Their approach demonstrated the effectiveness of using specialized features to difficult tagging decisions, and they proposed a training strategy mitigating the corpus consistency problems discovered during the implementation of specialized features. Rosenberg et al. [24] proposed the mixture-of-parents maximum entropy Markov model (MoP-MEMM), which is a class of directed graphical models extending MEMMs. Regardless of the range of the dependencies, their model can efficiently compute the exact marginal posterior node distributions.

2.3. CRF-based model

Lafferty et al. [5] used CRF for NLP, which is a type of statistical modeling method and is frequently applied in sequence labeling problems. The CRF offers several advantages over HMMs and stochastic grammars for such tasks, including the ability to relax strong independence assumptions made in those models. CRFs also avoid a fundamental limitation of maximum entropy Markov

models (MEMMs) and other discriminative Markov models using directed graphs, which are often biased toward states with few successor states. Others have applied variations of CRF to NLP tasks [25–27]. Tseng et al. [28] presented a Chinese word segmentation system using CRF to capture numerous linguistic features, such as character identity, morphology, and reduplication. Their system had no biases toward any particular variety of Mandarin and showed robust performance over different datasets. Zhao et al. [26] treated the Chinese word segmentation (CWS) problem as a character-based tagging problem with a CRF framework. Instead of focusing solely on feature template selection as in previous works, they considered both feature template selection and tag set selection. Zhao et al. [27] used a six-tag set together with tone features of Chinese characters and auxiliary segmenters trained by other corpora for further improving CRF-based Chinese word segmentation performance. The developed system achieved the highest F-measures on four tracks in the Third SIGHAN Chinese Language Processing Bakeoff. Cuong et al. [29] considered the problem of incorporating high-order dependencies between labels or segments into CRF. Under the assumption that the number of distinct label patterns in the features is small, they provided efficient inference algorithms to handle such problems. They showed experimentally that using high-order dependencies leads to substantial performance improvements for some problems.

Andrew [30] proposed a hybrid model capable of representing both CRF and semi-CRF features and described efficient algorithms for its training and inference. He demonstrated that the proposed hybrid model achieved error reductions of 18% and 25% over a standard first order CRF and a semi-CRF when segmenting Chinese words. Nguyen et al. [31] extended first-order semi-CRFs to include higher-order semi-Markov features and presented efficient inference and learning algorithms under the assumption that the higher-order semi-Markov features are sparse. They empirically demonstrated that high-order semi-CRFs outperformed high-order CRFs and first-order semi-CRFs on three sequence labeling tasks with long distance dependencies. Muis and Wei [25] proposed a weak semi-CRF approach to chunking of noun phrases. In conventional semi-CRF, the model intuitively determines the length and type of the next segment at the same time. In contrast, a weak semi-CRF model proposes a weaker variant that makes the two decisions separately by restricting each node to connect either to only the nodes of the same label in next segment or to all the nodes in the next word alone. The weak semi-CRF model yields performance similar to conventional semi-CRFs but runs significantly faster.

2.4. Deep learning-based model

Deep learning methods also have advantages in the task of sequence labeling. Some theoretical analysis methods are commonly used in this area [32–36]. Desmar surveyed the current works for the statistical analytics and then theoretically and empirically examined several suitable tests [35]. Garcia and Herrera [36] focused on the statistical procedures for comparing $n \times n$ classifiers. Shang et al. [33] presented a new deep memory convolution neural networks (M-Net) to alleviate the overfitting problem caused by insufficient SAR image samples. From the experimental results, the designed M-Net achieved higher accuracy than several well-known SAR image classification algorithms. A complex-valued convolutional autoencoder network (CV-CAE) [34] was further proposed to extend the encoding and decoding of convolutional autoencoder to complex domain, thus the phase information can be adopted. Meng et al. [32] developed a new semi-supervised graph regularized deep NMF with bi-orthogonal constraints (SGDNMF) that learns a representation from the hidden layers for clustering task. It incorporates dual-hypergraph Laplacian regularization, which can reinforce high-order relationships in both data and feature spaces and fully retain the intrinsic geometric structure of the original data. In addition to the statistical analytics of deep learning models, Huang et al. [37] proposed a variety of long short-term memory (LSTM) models for sequence labeling, including LSTM networks, bidirectional LSTM (Bi-LSTM) networks, LSTM with a CRF layer (LSTM-CRF), and bidirectional LSTM with a CRF layer (Bi-LSTM-CRF). Their work was the first to apply Bi-LSTM-CRF to NLP benchmark sequence prediction datasets, and it showed that the Bi-LSTM-CRF model efficiently used both past and future input features. Their model produces state-of-the-art accuracy on the part of speech tagging, chunking, and NER datasets, and its performance is less dependent on word embedding than previous approaches.

Semi-supervised non-negative matrix factorization (NMF) exploits the strengths of NMF in effectively learning local information contained in data and is also able to achieve effective learning when only a small fraction of data is labeled. NMF is particularly useful for dimensionality reduction of high-dimensional data. However, the mapping between the low-dimensional representation, learned by semi-supervised NMF, and the original high-dimensional data contains complex hierarchical and structural information, which is hard to extract by using only single-layer clustering methods. Therefore, in this article, we propose a new deep learning method, called semi-supervised graph regularized deep NMF with bi-orthogonal constraints (SGDNMF). SGDNMF learns a representation from the hidden layers of a deep network for clustering, which contains varied and unknown attributes. Bi-orthogonal constraints on two factor matrices are introduced into our SGDNMF model, which can make the solution unique and improve clustering performance. This improves the effect of dimensionality reduction because it only requires a small fraction of data to be labeled. In addition, SGDNMF incorporates dual-hypergraph Laplacian regularization, which can reinforce high-order relationships in both data and feature spaces and fully retain the intrinsic geometric structure of the original data. This article presents the details of the SGDNMF algorithm, including the objective function and the iterative updating rules. Empirical experiments on four different datasets demonstrate state-of-the-art performance of SGDNMF in comparison with six other prominent algorithms.

Dyer et al. [38] proposed a technique for learning representations of parser states in transition-based dependency parsers. Their primary innovation was a new control structure for sequence-to-sequence neural networks, the stack LSTM. As with conventional stack data structures used for transition-based parsing, elements can be pushed to or popped from the top of the stack constantly. Moreover, the LSTM maintains a continuous space embedding of the stack contents. This enables models to capture three facets of a parser's state: (i) an unbounded look-ahead into the buffer of incoming words; (ii) the complete history of actions taken by the parser; and (iii) the complete contents of the stack of partially built tree fragments, including their internal structures. Training uses standard backpropagation techniques and yields state-of-the-art parsing. Lample et al. [39] introduced two new neural architectures. The first uses Bi-LSTM and CRF, and the second constructs and labels segments using a transition-based approach, which is like shift-reduce parsers. Both models rely on two sources of word information: character-based word representations learned from the supervised corpus and unsupervised word representations learned from the unannotated corpora. These models achieved state-of-the-art performance on NER in four languages without resorting to any language-specific knowledge or resources.

Kong et al. [8] proposed a segmental recurrent neural network, which is a variant of semi-CRF. Given an input sequence,

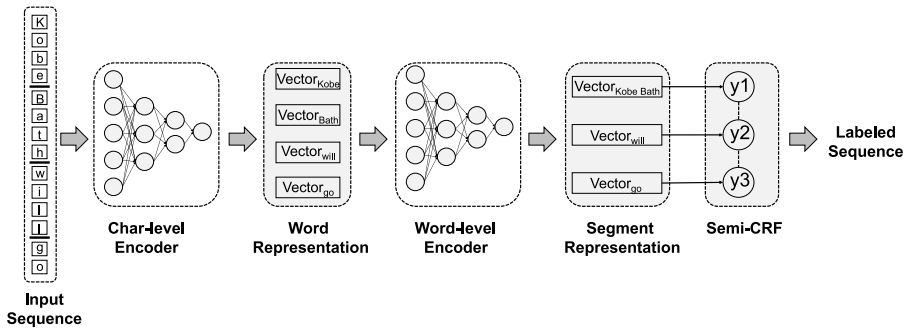

Fig. 1. The proposed framework.

a segmental recurrent neural network models a joint probability distribution over segmentations of the input and labels of the segments. This model shows higher performance compared to models that do not explicitly represent segments. Liu et al. [40] proposed a neural semi-CRF that composes both the pre-trained word embedding and pre-trained segment embedding. Zhuo et al. [41] proposed gated recursive semi-CRFs (grSemi-CRFs) that model segments directly and learn segment level features automatically through a gated recursive convolutional neural network (CNN). Ma and Hovy [9] proposed a CNN-LSTM-CRF model that benefits from both word-level and character-level representations automatically. Their system is end-to-end and requires no feature engineering or data preprocessing. Their evaluation with two datasets for part-of-speech tagging (POS tagging) and NER showed state-of-the-art performance on both datasets. Rei et al. [10] incorporated character-level information to handle the out-of-vocabulary (OOV) issue in sequence labeling and investigated character-level extensions to conventional LSTM-CRF structure models. They adopted another LSTM to encode character-level information. The encoded character-level information combines pre-trained word embeddings with an attention mechanism. By using an attention mechanism, the model is able to determine dynamically how much information to use from a word- or character-level component. They evaluated their models on a range of sequence labeling datasets and showed that the performance improvement was quite robust.

3. Preliminaries and problem statement

3.1. Semi-CRF

The semi-CRF models provide variable length segmentations of the label sequence. Compared with the traditional CRF, a semi-CRF model is capable of capturing more segment level information (i.e., boundary words features, segment length features, etc.). A semi-CRF architecture models the conditional probability of the possible out sequence s over input sequence x as

$$p(s|x) = \frac{1}{Z(x)} \exp\{W \cdot G(x, s)\}, \tag{1}$$

where $G(x, s)$ is the feature function, W is the weight vector, and $Z(x)$ is the normalization factor of all the possible segmentations s over x. To find the best segmentation in semi-CRF, let α_j denote the best segmentation ending with jth input; (m, n, y) denote a segment start at mth position; and it ends at the nth position and has the label y, α_j, calculated as

$$\alpha_j = \max_{l=1\ldots L, y} \psi(j-1, y) + \alpha_{j-l-1}, \tag{2}$$

where L is the maximum segment length and $\psi(j-1, y)$ is the feature value defined over segment $s = (j-l, j, y)$. The dynamic programming method can be adopted to recursively calculate the segment from length 1 to L.

3.2. Neural semi-CRF-based model

A neural semi-CRF-based neural network can efficiently use past input features through a neural network layer (i.e., LSTM) and sentence level tag transition information through a semi-CRF layer. A semi-CRF layer has a state transition matrix as a parameter. With such a layer, like using past and future input features within a LSTM network, the models have easy access to past and future tags for prediction of the current tag. The network outputs a matrix of scores, denoted as $F_\theta([x]_i^T)$. The element $[F_\theta]_{i,t}$ of the matrix $(F_\theta([x]_i^T))$ is the score outputted by the network with parameters θ for the sentence $[x]_1^T$ and the ith tag at the tth word. The transition score $[A]_{i,j}$ models the transition from the ith state to the jth state for consecutive time steps. It should be noted that this transition matrix is position independent. The score of a sentence $[x]_i^T$ along with a path of tags $[i]_i^T$ is then given by the sum of transition scores and network scores as

$$s([x]_i^T, [i]_i^T, \Theta) = \Sigma_{t=1}^T([A]_{[i]_{t-1},[i]_t} + [F_\theta]_{[i]_t,t}). \tag{3}$$

3.3. Problem statement

Formally, given an input sequence $x = (x_1, \ldots, x_k)$ of length k, let $x_{a:b}$ denote the subsequence $(x_a, \ldots, x_b)$, where $a \leq b \leq k$. A segment of x is defined as a triad (u, v, y) that associates the subsequence $x_{u:v}$ with label y. A segmentation of x is a segment sequence $s = (s_1, \ldots, s_p)$, where $s_j = (u_j, v_j, y_j)$ and $u_{j+1} = v_j + 1$. Given an input sequence x, the segmental labeling problem is defined as the problem of finding x's most probable segment sequence s.

4. Proposed models for sequence prediction

This section introduces the proposed hierarchical attention neural semi-Markov random fields model. The proposed model shown in Fig. 1, and it has three layers as, a character-level encoder, a word-level encoder, and a semi-CRF layer, from the bottom to the top. The bottom character-level encoder is used to extract character-level information of each word. The extracted character-level information (i.e., a n-dimension vectors, where n is a hyper parameter) is concatenated with the word embedding vectors to form the input of the word-level encoder. The word-level encoder is then used to extract word-level information from each concatenated input vector and compute the neural feature score of each label. Finally, the computed feature scores are sent to the semi-CRF layer and combined with the conventional semi-CRF layer feature scores to jointly train the semi-CRF model. These three layers are introduced in detail below.

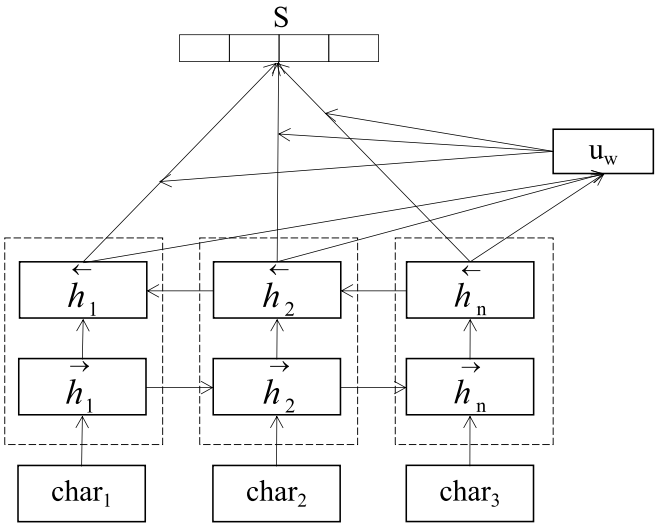

Fig. 2. Attention mechanism.

4.1. Neural network for character-level representation

In the bottom layer, a neural network encoder is used to incorporate character-level information into the neural network. This has two primary advantages compared to word-level information. First, character-level information is inherently useful for modeling out-of-vocabulary (OOV) words. For sequence labeling tasks, like named entity recognition (NER), it is quite common for entities, such as the organization or person, to be unseen words. Second, character-level information provides external morphological information from the characters of words; for example, suffixes, such as "ing" and "ed", are important markers of adjective words when tagging parts-of-speech. The character-level encoder is then constructed in the designed model, which is shown in Fig. 3. This encoder uses the bottom Bi-LSTM to obtain the context vector (i.e., c_i) from the input sequences and then uses the top Bi-LSTM to calculate the representation for each segment (i.e., word). Each token is represented as the concatenation of a forward and backward direction of the bottom Bi-LSTM over the sequence of raw character inputs. The ASRNN uses this representation rather than directly reading the embeddings. This permits tokens to be sensitive to the contexts where they occur and is typical of neural network sequence prediction models [8]. After computing the representations of each token (i.e., character) using the bottom Bi-LSTM, the top Bi-LSTM is used to compute representations for each span (i.e., word). Rather than the approach proposed by Kong et al. [8] in 2016, which directly concatenates the final states of two directions in the Bi-LSTM, an attention mechanism like prior work [42] is used to combine the outputs of all the timesteps. An illustration is provided in Fig. 2, and the formulas are defined as follows:

$$u_{it} = tanh(W_w h_{it} + b_w), \tag{4}$$

and

$$\alpha_{it} = \frac{exp(u_{it}^T u_w)}{\Sigma exp(u_{it}^T u_w)}, \text{ and } s_i = \Sigma_t \alpha_{it} h_{it}. \tag{5}$$

Specifically, we first concatenate the output of the forward and backward LSTM to form the h_{it}, which denotes the output of the Bi-LSTM of the ith sequence at the t-timestep. Then, each h_{it} is fed through a single-layer fully connected network to obtain u_{it} as a hidden representation of the current t-timestep, and the importance of the current character is computed as the similarity of u_{it} with a character level global vector u_w. The normalized importance weight is obtained by the *softmax* function. After that, the word representation s is computed as a weighted sum of the hidden representations h_{it} at each timestep based on the corresponding weight α_{it}. The final word representation sequence s_i of the input sequence is shown at the top of Fig. 3.

4.2. ASRNN for word-level representation

The ASRNN model is used for the word level encoder, which is shown in Fig. 4. The extracted character-level representations are concatenated with the corresponding pre-trained glove word embeddings [43]. A Bi-LSTM is performed to obtain the context vector c_k at each timestep k and another Bi-LSTM is recursively performed to dynamically calculate the segmental representation seg_i for each segment i from the context vector c_k with an attention mechanism. The computation progress is a dynamic recursion, which computes the segment from length 1 to L, where L is the maximum segment length. A fully connected layer is then used to perform the label classification on each segmental representation seg_i. The label score computed by a fully connected layer is directly used as the neural feature scores. It should be noted that it is unnecessary to perform the *softmax* operation for label classification with the fully connected layer since the sum of the neural feature scores can be larger than 1. Finally, the computed neural feature scores are transferred to the semi-CRF model, and the conventional semi-CRF features are combined to jointly train the semi-CRF model. The computation process is given below.

$$c_{it} = [forward_{LSTM_1}(x) : backward_{LSTM_1}(x)], \tag{6}$$

$$h_{it} = [forward_{LSTM_2}(c_{it}) : backward_{LSTM_2}(c_{it})], \tag{7}$$

and

$$s_i = \Sigma_t \alpha_{it} h_{it}, \text{ and } F_i = MLP(s_i), \tag{8}$$

where x is the input as the concatenation of the pre-trained word embedding and the character-level information, [:] is the concatenate operation, c_{it} is the context vector, α_{it} is the attention weight computed by formula in Eq. (5), and MLP is a fully connected neural network used as the projection layer of the Bi-LSTM neural network to compute the neural feature scores.

It should be noted that both the character-level and word-level encoders utilize an attention mechanism, which enables the model to attend differently to different characters and words. This information is quite important when doing sequence labeling tasks. Character level information, such as suffixes "ing" and "ed", indicates that the word tends to be an adjective when in the POS tag task. Word level information, such as "is", indicates that the front segment is an NP in the NER task. Here, an attention mechanism is used in the RNN to help the model extract such information and further enhance its performance. For the attention based encoder model, the time complexity is $O(N^2)$, where N is the length of the input sentence. Although the time complexity of an attention based encoder is larger than conventional RNN ($O(N)$), the main concern in this paper is on the performance gain.

4.3. The semi-CRF layer

The semi-CRF layer is depicted at the top of Fig. 4, where seg_i is the segmental representations extracted by neural networks, and the circle nodes y_1, y_2 comprise the observed output sequences. Dashed lines indicate optional features that can be included in the semi-CRF. Following prior work [37], we considered only the label transition in this work. The objective function is modified to attend differentially to neural features and conventional CRF sparse

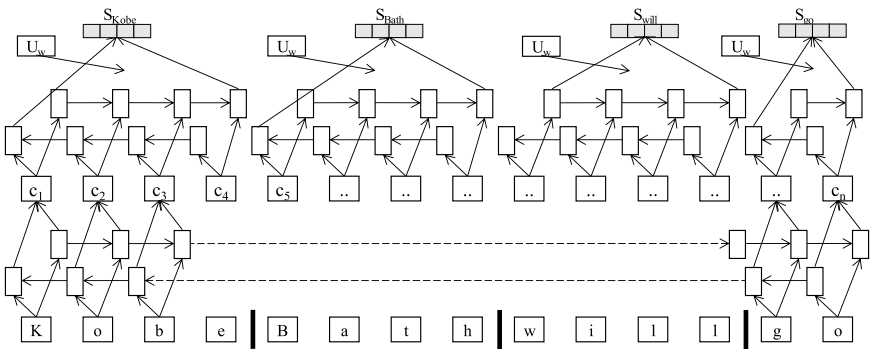

Fig. 3. Character-level ASRNN encoder.

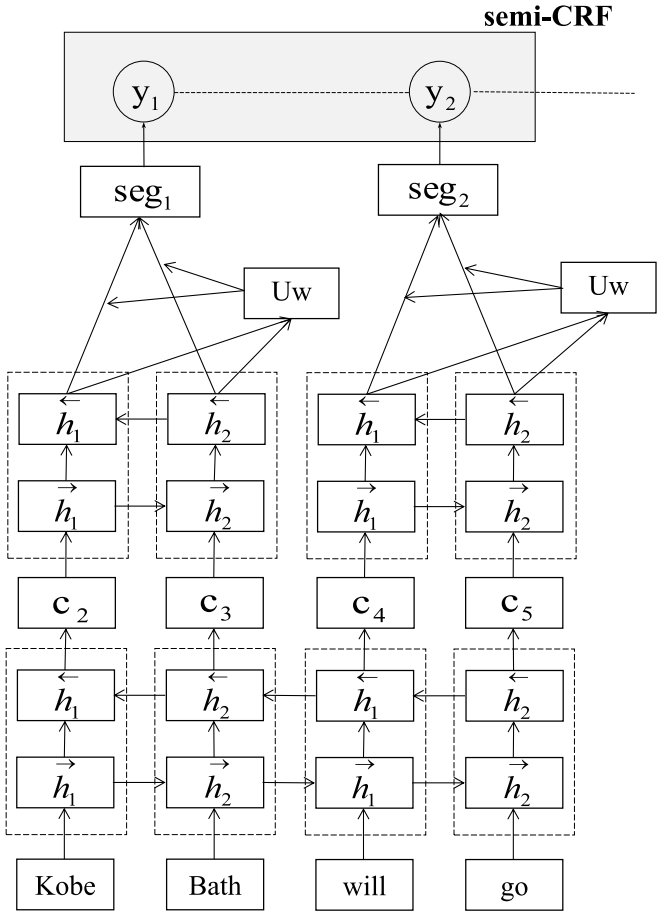

Fig. 4. Word-level ASRNN encoder.

features. The modified semi-CRF layer models the conditional probability of possible output sequence s over input sequence x as follows:

$$p(s|x) = \frac{1}{Z(x)} exp\{w_1 A(x,s) + w_2 F(x,s)\}, \tag{9}$$

where $A(x,s)$ is the conventional semi-CRF feature scores, w_1 and w_2 are the corresponding weights of the sparse CRF features and neural features, $F(x,s)$ is the set of neural feature scores computed by the proposed hierarchical attention neural network, and $Z(x)$ is the normalization factor of all the possible segmentations s over x. For training our neural semi-CRF, we used maximum conditional likelihood estimation. For a training set $\{(x_i, s_i)\}$, the log-likelihood is given as

$$L_D(W) = \Sigma_{i \in D} log\, p(s|x). \tag{10}$$

The time complexity of the semi-CRF layer is $O(nLY^2)$, where n is the sequence length, L is the segment length, and Y is the number of labels.

4.4. Parameter settings

Here, we briefly introduce the parameter settings utilized in this study.

- **Word Embeddings:** We used Stanford Glove 100-dimension word embeddings, which were trained from 6 billion Wikipedia data and web texts, and other pre-trained word embeddings were also used.
- **Character Embeddings.** We randomly initialized character embeddings with uniform samples from $\left[\sqrt{\frac{3}{dim}}, \sqrt{\frac{3}{dim}}\right]$.
- **Optimization Algorithm:** We used mini-batch stochastic gradient descent with batch size of 10 and a momentum of 0.9. The initial learning rate is set as 0.01 with decay rate 0.1. Gradient clipping was set to 5 to prevent the "gradient exploding". We also used a dropout rate of 0.5 during the training progress to prevent overfitting.
- **Network Structure:** For character level encoder, the hidden dimension of both the bottom Bi-LSTM network and the upper Bi-LSTM was set to 50, which yielded a 100-dimension character-level representation vector. For word-level encoder, the hidden dimension of both the bottom Bi-LSTM network and the upper network was set to 100, which yielded a 200-dimension segmental representation. The fully connected layer had a hidden size of 256, and the output size was equal to the number of labels at each task.

For most of the experiments, the pre-trained word embedding was important and yielded a much better performance than random initialized word embeddings. It should be noted that other word embeddings, such as word2vec, worked fine with the proposed model. We also found that the performance was not sensitive to the optimization algorithm when using pre-trained character embeddings. The characters were quite limited compared to the words, and thus we directly used a randomly initialized character vector.

4.5. An illustrated example

A simple example is given here to show the process of the developed neural network. For simplicity, an NER task with only two labels is used. Let N denote a name entity of a segment, and O be not a name entity. From a given string of "Kobe Bath

will go", the name entity is then extracted. In this example, "Kobe Bath" is a name entity, obviously, while "will" and "go" are not the name entity. The character-level encoder is first used to extract character-level information of the given sequences shown in Fig. 3. The character-level information is especially useful while handling out-of-vocabulary (OOV) words, such as "Kobe" and "Bath". The character embedding is sent to a Bi-LSTM model, and the output of the forward LSTM and backward LSTM are concatenated into the context vector c_i, which is the dark rectangle shown in Fig. 3. For each word, a shared-parameter of the Bi-LSTM model is used to extract the character-level word representation from the context vector c_i. For example, the word "Kobe" can be represented as four context vectors such as c_1, c_2, c_3, c_4, and they are used as inputs of the model. The outputs of the forward LSTM and backward LSTM at each timestep are concatenated together as the output of current timestep. Thus, we can obtain four output vectors (respectively denote as o_1 to o_4) corresponding to four characters of a word "Kobe". The attention mechanism calculates the weighted of the character-level word represented by $S_{Kobe} = \alpha_1 c_1 + \alpha_2 c_2 + \alpha_3 c_3 + \alpha_4 c_4$, where α_i is the attention weight computed of each timestep. The same processing is then performed to every word of the sequence, and the character-level information can be successfully extracted as S. The extracted S is then combined with the pre-trained word embeddings (i.e., concatenate S_{Kobe} with word embedding of "Kobe") to form as the input of the word-level encoder, which is shown as the bottom rectangle of Fig. 4. Currently, we have obtained four context vectors such as c_1, c_2, c_3, and c_4, and those content vectors are used to extract and classify all the possible segments. For example, in Fig. 4, a Bi-LSTM model with attention mechanism is used to extract features from two segments (i.e., "Kobe Bath" and "will go" in this example). A fully connected neural network then classifies whether the segment belongs to the N or O label. At this stage, we can obtain the estimated probabilities of N (name entity) or O (not name entity) for the "Kobe Bath" and "will go". This progress is similar as the emission feature in the CRF model. Thus, for the segment "Kobe Bath", it would obtain a very large score of the label N since it is classified as a name entity, and its score of label O would be very small. After that, the dynamic programming method [8] is used to perform the same progress (extracting features and classification) on all possible segments. As the mentioned above, each segment obtains two scores corresponding to the probability of N or O. The obtained scores form as the neural features in our designed model, which have the same form with the conventional features in CRF/semi-CRF models. After that, the neural features are combined with the features (i.e., transition features) of semi-CRF to jointly decode the best label of the input sequence by Viterbi algorithm [44].

5. Experimental evaluation

We evaluated our ASRNN model on three NLP tasks of sequence labeling including: named entity recognition (NER), chunking, and reference parsing.

5.1. Compared models

Three conventional statistic sequence labeling methods are considered, namely sparse CRF, semi-CRF and MEM models. These models directly use hand-craft sparse features. Three neural based models are considered, namely SRNN, NSML, and BERT. Deep learning based models can extract features automatically. Compared with above methods, the proposed methods can be effectively combine hand-craft features and automatically extract neural features. It can also balance these two types of features by a tunable weights.

Table 1
Statistics of datasets.

Dataset	Task	#labels	#trains	#dev	#test
CoNLL00	Chunking	22	8,936	N/A	2,012
CoNLL02	NER	8	14,987	3,466	3,684
Cora	Ref Parsing	13	300	100	100
BC2GM	NER	3	12,500	2,500	5,000
JNLPBA	NER	11	18,546	N/A	3,856

Table 2
A snippet of data.

1. U.N./ORG official/O Ekeus/PER heads/O for/O Baghdad/LOC
2. That/NP 's/VP the/NP dilemma/NP for/PP today/NP 's/NP parent/NP
3. Horn,/A B./A (1986)./Y Robot/T Vision,/T MIT/P Press./P

- **Sparse CRF:** The CRF model using sparse hand-crafted features. For the NER and Chunking tasks, we used the **baseline feature template** from Guo et al. in 2014 [45]. For the reference parsing task, we compared with the state-of-the-art reference parsing software Parscit in 2008 [46] that uses a CRF model with several hand-crafted features, task-specific features.
- **Sparse semi-CRF:** The semi-CRF model [7] using sparse hand-crafted features. Features defined in the semi-CRF are exactly the same as the one used in the sparse CRF models.
- **MEM:** Maximum entropy model (MEM) is a maximum-likelihood approach for automatically constructing maximum-entropy models, similar sparse features are utilized in MEM models.
- **SRNN:** The Segmental recurrent neural network proposed by Kong et al. in 2016 [8].
- **Neural sequence labeling model(NSML):** A hierarchical LSTM–LSTM-CRF neural network with a specially designed attention mechanism proposed by Rei et al. in 2016 [10]. The designed attention mechanism is used to perform a weighted sum to combine extracted character-level representations and the pre-trained word embeddings instead of directly performing concatenation.
- **BERT:** A pre-trained language model proposed by Devlin et al. in 2018 [47].

5.2. Dataset description

We then used the following datasets for the experimental evaluation. The used datasets in this paper are commonly conducted in the NLP fields, which are the public datasets and can be accessed from [48–50]. Precision(P), recall(R) and f1 score(F) are used as evaluation matrix in all our experiments. The corpora statistics are shown in Table 1 and a snippet of the data is shown in Table 2. The form of the used data is represented as [word/label], thus each word is marked with a corresponding label (i.e., ORG, PER and LOC, among others). Details of the used datasets are given as below.

- **CoNLL2003.** We performed experiments on the English data from CoNLL 2003 shared task [49] for Named entity recognition. This dataset contains four different types of named entities: person, location, organization, and miscellaneous.
- **BC2GM.** We performed experiments on the BioCreative II Gene Mention corpus [51] consisting of 20,000 sentences from biomedical publication abstracts with annotations for mentions of the names of genes, proteins, and related entities using a single NE class.
- **JNLPBA.** We performed experiments on the JNLPBA corpus [41] which consists of 2,404 biomedical abstracts annotated for mentions of five entity types: cell line, cell

type, DNA, RNA, and protein. The corpus was derived from GENIA corpus entity annotations for use in the shared task organized in conjunction with the BioNLP 2004 workshop.

- **CHEMDNER.** We performed experiments on the BioCreative IV Chemical and Drug NER corpus [52] that consists of 10,000 abstracts annotated for mentions of chemical and drug names using a single class.
- **CoNLL2000.** We performed experiments on the English data from CoNLL 2000 shared task [48] for the chunking task. We used Wall Street Journal Sections 15–18 from the Penn Treebank for training and Section 20 as the test data.
- **Cora.** We performed experiments with the Cora dataset for reference parsing. Cora [50] contains 500 reference strings labeled by 13 fields, including author, title, book title, journal, volume, pages, note, tech, date, editor, location, institution, and publisher.

5.3. Named Entity Recognition (NER)

Named entity recognition (NER) is a task of locating and classifying named entities in text into the pre-defined categories, such as the names of persons, organizations, or locations, among others. Tables 3–6 show the performance of the test models in the NER task on the CoNLL2003, BC2GM, CHEMDNER, and JNLPBA datasets. The best performance is marked with an underline. The "the proposed model" is with the standard ASRNN character-level encoder. The proposed ASRNN encoder outperforms the other models, while the proposed ASRNN encoder always obtains the highest F score. It can be observed that the conventional methods (MEM, CRF, semi-CRF) performs poorer on most datasets, this denotes that the performance of these types of models are largely depend on the hand-craft sparse features. These sparse features are commonly varies between tasks and datasets. The proposed ASRNN uses attention mechanism to dynamically extract features for different tasks and datasets. Compared with word-level neural models (NSLM and BERT), the ASRNN captures more character level features. The proposed model achieves better performance which denotes the fact that the character-level information is essential for sequence labeling tasks and offer better handling of out-of-vocabulary (OOV) problems. The OOV problem should be carefully handled in the NER task. For an entity like person name, it is very sparse or even does not appear in the training data; thus, hand-crafted features are required to handle this problem. The designed model can, however, use character-level information to solve this issue. A visualization of attention weight computed by the character-level encoder is shown in Fig. 5. The darker the rectangle represents a higher attention weight. As shown, the capital character attracts more attention by the designed model. Those capital characters show a strong indication of a person's name. Although the person's name may not exists in the training data, the model can still utilize the attention mechanism to label it correctly with character level information. Thus, it can be concluded that the designed model can utilize an attention mechanism to reveal the person's name even if it did not appear in the training data. Compared with the proposed model, the conventional models (MEM, sparse CRF, and sparse semi-CRF) performed poorly on this dataset as the performance is largely dependent on feature engineering. This also implies that it is difficult to design the specific features to handle the OOV problem.

5.4. Chunking

Chunking is a text processing method to identify the constituent parts of texts (nouns, verbs, or adjectives, among others) and then combine the texts into higher order units that have discrete grammatical meanings (noun groups or phrases, or verb groups, among others). Table 7 shows the performance of the compared models on the chunking CoNLL2000 shared task [48]. Again, the proposed models outperformed the others. The performance of the semi-CRF was close to the CRF model. This indicates that the performance of the CRF-based models are largely dependent on the feature engineering. Although the semi-CRF model could directly model the segments of the sequences better than the conventional CRF model, the performance of those two models was similar. The reason for this is that those two models used exactly the same features in the experiments without any feature engineering. Compared with the conventional CRF and semi-CRF models, the neural CRF-based and semi-CRF-based models (Rei et al. in 2016, Kong et al. in 2016 and Devlin et al. in 2018, respectively) had great improvements. This shows that the neural network models can automatically capture features instead of hand-crafted features.

Table 3
Comparison of results on the CoNLL2003 dataset.

NER	P	R	F
MEM	84.20	81.33	82.74
Sparse CRF	84.10	83.59	83.84
Sparse semi-CRF	83.82	84.36	84.09
NSLM 2016	84.15	85.23	84.69
SRNN 2016	84.21	86.35	85.26
BERT 2018	90.8	**92.0**	91.4
Proposed ASRNN	**93.2**	90.7	**92.0**

Table 4
Results on BC2GM.

NER	P	R	F
MEM	85.94	87.62	86.77
Sparse CRF	86.50	87.88	87.18
Sparse semi-CRF	86.88	88.05	87.46
NSLM 2016	87.91	87.98	87.94
SRNN 2016	88.02	88.10	88.05
BERT 2018	**91.5**	89.8	90.6
Proposed ASRNN	**91.5**	**90.1**	**90.8**

Table 5
Comparison of results on the JNLPBA dataset.

NER	P	R	F
MEM	71.16	70.42	70.78
Sparse CRF	70.23	71.18	70.70
Sparse semi-CRF	71.01	70.88	70.94
NSLM 2016	71.59	71.01	71.30
SRNN 2016	71.68	71.22	71.45
BERT 2018	73.3	74.2	73.7
Proposed ASRNN	**73.7**	**74.4**	**74.0**

Table 6
Comparison of results on the CHEMDNER dataset.

NER	P	R	F
MEM	81.69	82.92	82.30
Sparse CRF	82.21	83.92	83.05
Sparse semi-CRF	82.62	84.16	83.38
NSLM 2016	83.48	85.56	84.50
SRNN 2016	83.59	84.86	84.22
BERT 2018	**85.1**	84.3	84.7
Proposed ASRNN encoder	85.0	**85.8**	**85.4**

5.5. Reference parsing

Reference parsing is a task of extracting information, such as authors, title, year, or journal, from a reference string. Table 8 shows a performance comparison for reference parsing on the Cora dataset [50]. As shown, the proposed models achieved best

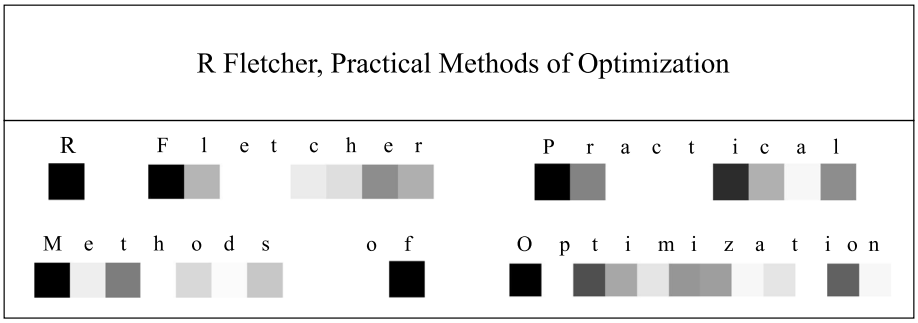

Fig. 5. Visualization results of attention mechanism.

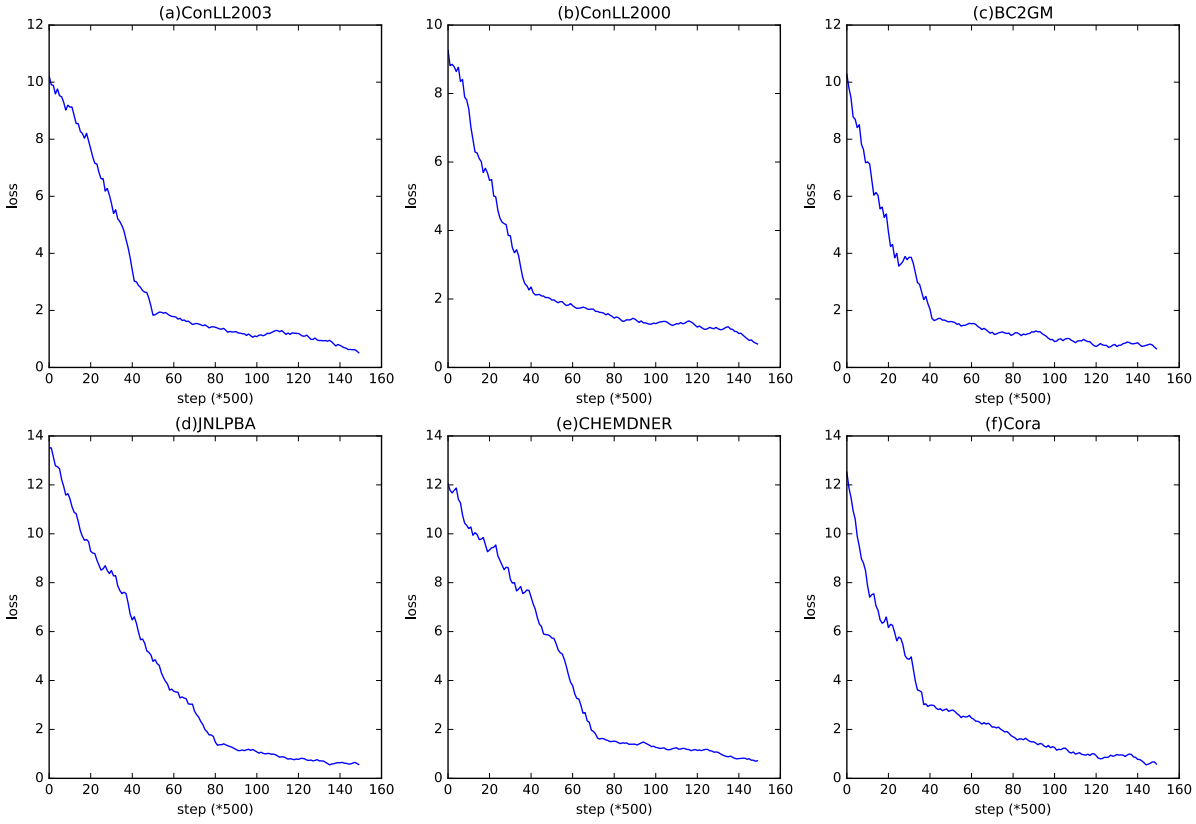

Fig. 6. Loss curve of the training process.

Table 7
Comparison of results on the CoNLL2000 dataset.

Chunking	P	R	F
MEM	90.21	88.57	89.43
Sparse CRF	90.05	89.88	89.96
Sparse semi-CRF	90.25	90.12	90.18
NSLM 2016	92.86	92.15	92.50
SRNN 2016	91.51	91.67	91.59
BERT 2018	**94.2**	92.4	93.3
Proposed ASRNN	93.72	**93.69**	**93.70**

Table 8
Comparison of results on the Cora dataset.

Reference Parsing	P	R	F
MEM	77.97	81.00	79.45
Sparse CRF	77.92	80.61	79.24
Sparse semi-CRF	78.35	81.21	79.75
NSLM 2016	77.75	80.12	78.91
SRNN 2016	76.53	78.44	77.47
BERT 2018	**78.60**	81.5	80.0
Proposed ASRNN	78.50	**81.95**	**80.18**

performance. Compared with the other two tasks, reference parsing had more segmental features. The segments, such as title and author, in reference parsing are normally longer than that in the other two tasks. Those segments involve the formal structure. For example, the initial character of the words in the segment of the authors is the capital character, and each author can be split by comma or semi-common. Since the proposed models directly model the segment rather than words, they can capture his information more effectively. The proposed models could achieve robust performance on reference parsing without any feature engineering. As shown in Fig. 5, the attention mechanism played important role on the capital words than the others for the character-level encoder. In a change from the experiments on the CoNLL2000 and CoNLL2003 datasets, the performance of sparse CRF and sparse semi-CRF improved greatly on this dataset, and they performed similarly to our proposed models. This is because

both models use the task-specific features proposed by Councill et al. [46], including token identity, N-gram prefix/suffix, orthographic case, and punctuation, among others. This also indicates that the performance of the conventional CRF-based methods largely depends on task-specific feature engineering. In contrast, the proposed neural semi-CRF model requires no feature engineering, such as hand-crafted features or task-specific knowledge, but can still achieve robust performance over several different sequence labeling tasks. A loss curve of the designed model on each task is also shown in Fig. 6. The x axis is the running steps and the y axis is the loss value. As shown, the designed model converged, which is to say it remained stable on all the tasks. For instance, the designed model converged around 20,000 on the BCG2M dataset.

6. Conclusion and future work

In this paper, we have presented the problem of incorporating character-level information into a neural semi-CRF model and proposed an attention segmental recurrent neural network (ASRNN) based on the hierarchical attention neural semi-CRF model for the task of sequence labeling. In the conventional CRF-based model, character-level information, such as prefix and suffix features, have been shown to be quite effective. The proposed ASRNN model extracts these features automatically. Empirical results from several sequence labeling tasks show that all two encoders offered robust performance over different sequence labeling tasks. The proposed model benefits from the hierarchical structure and achieves competitive performance with a variety of data. There are several directions for this work to be extended in the future, such as exploring more variants of an attention mechanism and handling other information extraction tasks (i.e., non-consecutive sequence labeling and fully character-level sequence labeling) and exploring different kind of attention mechanisms, such as separating the attention mechanism for two directions of Bi-LSTM.

CRediT authorship contribution statement

Jerry Chun-Wei Lin: Development of conceptualization, Methodology, Formal analysis. **Yinan Shao:** Development of conceptualization, Methodology, Formal analysis. **Youcef Djenouri:** Experimental validation. **Unil Yun:** Formal review and editing.

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.

References

[1] P. Gupta, B. Andrassy, Table filling multi-task recurrent neural network for joint entity and relation extraction, in: International Conference on Computational Linguistics, 2016, pp. 2537–2547.

[2] M. Mintz, S. Bills, R. Snow, D. Jurafsky, Distant supervision for relation extraction without labeled data, in: International Joint Conference on Natural Language Processing, 2009, pp. 1003–1011.

[3] S. Guo, M.W. Chang, E. Kiciman, To link or not to link? a study on end-to-end Tweet entity linking, in: Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2013.

[4] J. Lu, D. Venugopal, V. Gogate, V. Ng, Joint inference for event coreference resolution, in: International Committee on Computational Linguistics, 2016, pp. 3264–3275.

[5] J.D. Lafferty, A. Mccallum, F.C.N. Pereira, Conditional random fields: probabilistic models for segmenting and labeling sequence data, in: The International Conference on Machine Learning, 2001, pp. 282–289.

[6] A.L. Berger, S.A.D. Pietra, V.J.D. Pietra, A maximum entropy approach to natural language processing, Comput. Linguist. 22 (1) (1996) 39–71.

[7] S. Sarawagi, W.W. Cohen, Semi-Markov conditional random fields for information extraction, in: The Annual Conference on Neural Information Processing Systems, 2004, pp. 1185–1192.

[8] L. Kong, C. Dyer, N.A. Smith, Segmental recurrent neural networks, in: The International Conference on Learning Representations, 2016.

[9] X. Ma, E. Hovy, End-To-End Sequence Labeling Via Bi-Directional LSTM-CNNs-CRF, The Association for Computational Linguistics, 2016, pp. 1064–1074.

[10] M. Rei, G.K.O. Crichton, S. Pyysalo, Attending to characters in neural sequence labeling models, arXiv:1611.04361, 2016.

[11] J. Zhuo, Y. Cao, J. Zhu, B. Zhang, Z. Nie, Segment-level Sequence Modeling Using Gated Recursive Semi-Markov Conditional Random Fields, Association for Computational Linguistics, 2016, pp. 1413–1423.

[12] L.E. Baum, T. Petrie, Statistical inference for probabilistic functions of finite state Markov chains, Ann. Math. Stat. 37 (6) (1996) 1554–1563.

[13] L.E. Baum, J.A. Eagon, An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology, Bull. Amer. Math. Soc. 37 (3) (1967) 360–363.

[14] L.E. Baum, G.R. Sell, Growth transformations for functions on manifolds, Pacific J. Math. 27 (2) (1968) 211–227.

[15] L.E. Baum, T. Petrie, G. Soules, N. Weiss, A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains, Ann. Math. Stat. 41 (1) (1970) 164–171.

[16] L.E. Baum, An inequality and associated maximization technique in statistical estimation of probabilistic functions of a Markov process, Inequalities 3 (1972) 1–8.

[17] S. Fine, Y. Singer, N. Tishby, The Hierarchical Hidden Markov Model: Analysis and Applications, Kluwer Academic Publishers, 1998.

[18] H.P. Zhang, Q. Liu, X.Q. Cheng, H. Zhang, H.K. Yu, Chinese lexical analysis using hierarchical hidden Markov model, The SIGHAN Workshop on Chinese Language Processing, vol. 17 (8), 2003, pp. 63–70.

[19] D. Shen, J. Zhang, G. Zhou, J. Su, C.L. Tan, Effective adaptation of a hidden Markov model-based named entity recognizer for biomedical domain, in: The Association for Computational Linguistics workshop on Natural Language Processing in Biomedicine, 2003, pp. 49–56.

[20] J.H. Lim, Y.S. Hwang, S.Y. Park, H.C. Rim, Semantic role labeling using maximum entropy model, in: The Conference on Computational Natural Language Learning, 2004, pp. 1–4.

[21] W. Sun, H. Li, Z. Sui, The Integration of Dependency Relation Classification and Semantic Role Labeling Using Bilayer Maximum Entropy Markov Models, The Association for Computational Linguistics, 2008, pp. 243–247.

[22] D. Yu, L. Deng, A. Acero, Using continuous features in the maximum entropy model, Pattern Recognit. Lett. 30 (14) (2009) 1295–1300.

[23] A. Ratnaparkhi, A maximum entropy model for part-of-speech tagging, in: The Conference on Empirical Methods in Natural Language Processing, 1996, pp. 133–142.

[24] D.S. Rosenberg, K. Dan, B. Taskar, Mixture-of-Parents Maximum Entropy Markov Models, The Association for Uncertainty in Artificial Intelligence, 2007, pp. 318–325.

[25] A.O. Muis, W. Lu, Weak semi-Markov CRFs for noun phrase chunking in informal text, in: The Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2016, pp. 714–719.

[26] H. Zhao, C.N. Huang, M. Li, T. Kudo, An improved Chinese word segmentation system with conditional random field, in: The SIGHAN Workshop on Chinese Language Processing, 2006, pp. 162–165.

[27] H. Zhao, C.N. Huang, M. Li, B.L. Lu, Effective tag set selection in Chinese word segmentation via conditional random field modeling, in: Pacific Asia Conference on Language, Information and Computation, 2006, pp. 87–94.

[28] H. Tseng, P. Chang, G. Andrew, D. Jurafsky, C. Manning, A conditional random field word segmenter for sighan bakeoff 2005, in: The SIGHAN Workshop on Chinese Language Processing, 2015, pp. 168–171.

[29] N.V. Cuong, N. Ye, W.S. Lee, L.C. Hai, Conditional random field with high-order dependencies for sequence labeling and segmentation, J. Mach. Learn. Res. 15 (1) (2014) 981–1009.

[30] G. Andrew, A hybrid Markov/semi-Markov conditional random field for sequence segmentation, in: Conference on Empirical Methods in Natural Language Processing, 2006, pp. 465–472.

[31] V.C. Nguyen, N. Ye, W.S. Lee, L.C. Hai, Semi-Markov conditional random field with high-order features, J. Mach. Learn. Res. 15 (1) (2014) 981–1009.

[32] M. Yang, R.H. Shang, F.H. Shang, Semi-supervised graph regularized deep NMF with bi-orthogonal constraints for data representation, IEEE Trans. Neural Netw. Learn. Syst. (2019) 1–14.

[33] R.H. Shang, J.M. Wang, L.C. Jiao, SAR Targets classification based on deep memory convolution neural networks and transfer parameters, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 11 (8) (2018) 2834–2846.

[34] R.H. Shang, G.G. Wang, A.O. Michael, Complex-valued convolutional autoencoder and spatial pixel-squares refinement for polarimetric SAR image classification, Remote Sens. 11 (5) (2019) 1–19.

[35] J. Demsar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. (2006) 1–30.
[36] S. Garcí a, F. Herrera, An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons, J. Mach. Learn. Res. (2008) 2677–2694.
[37] Z. Huang, W. Xu, K. Yu, Bidirectional LSTM-CRF models for sequence tagging, in: Conference on Empirical Methods in Natural Language Processing, https://arxiv.org/abs/1508.01991, 2015.
[38] C. Dyer, M. Ballesteros, W. Ling, A. Matthews, N.A. Smith, Transition Based Dependency Parsing with Stack Long Short Term Memory, Association for Computational Linguistics, 2015, pp. 334–343.
[39] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, C. Dyer, Neural Architectures for Named Entity Recognition, North American Chapter of the Association for Computational Linguistics, 2016, pp. 260–270.
[40] Y. Liu, W. Che, J. Guo, Q. Bin, T. Liu, Exploring segment representations for neural segmentation models, in: International Joint Conference on Artificial Intelligence, 2016, pp. 2880–2886.
[41] J. Kim, T. Ohta, Y. Tsuruoka, Y. Tateisi, N. Collier, Introduction to the bio-entity recognition task at JNLPBA, in: Joint Workshop on Natural Language Processing in Biomedicine and its Applications, 2004, pp. 70–75.
[42] Z. Yang, D. Yang, C. Dyer, A. Smola, E. Hovy, Hierarchical attention networks for document classification, in: The Annual Conference of the North American Chapter of the Association for Computational Linguistics, 2016, pp. 1480–1489.
[43] J. Devlin, M.W. Chang, K. Lee, T. Kristina, BERT: Pre-training of deep bidirectional transformers for language understanding, arXiv:1810.04805, 2018.
[44] A.J. Viterbi, Error bounds for convolutional codes and an asymptotically optimum decoding algorithm, IEEE Trans. Inform. Theory 13 (2) (1967) 260–269.
[45] J. Guo, W. Che, Hai Wang, T. Liu, Revisiting embedding features for simple semi-supervised learning, in: Conference on Empirical Methods in Natural Language Processing, 2014, pp. 110–120.
[46] I.G. Councill, C.L. Giles, M.Y. Kan, ParsCit: an open-source CRF reference string parsing package, in: The International Conference on Language Resources and Evaluation, 2008, pp. 661–667.
[47] M.E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, L. Zettlemoyer, Deep Contextualized Word Representations, The North American Chapter of the Association for Computational Linguistics, 2019, pp. 2227–2237.
[48] E.F.T.K. Sang, F.D. Meulder, Introduction to the CoNLL-2000 Shared Task: Chunking, The Association for Computational Linguistics, 2000, pp. 127–132.
[49] E.F.T.K. Sang, F.D. Meulder, Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition, The Association for Computational Linguistics, 2003, pp. 142–147.
[50] K. Seymore, A. McCallum, R. Rosenfeld, Learning Hidden Mmodel Structure for Information Extraction, The Association for the Advancement of Artificial Intelligence, 1999, pp. 37–42.
[51] L. Smith, L.K. Tanabe, R.J. n. Ando, C. Kuo, I. Chung, C. Hsu, Y. Lin, R. Klinger, C.M. Friedrich, K. Ganchev, M. Torii, H. Liu, B. Haddow, C.A. Struble, R.J. Povinelli, A. Vlachos, W.A. Baumgartner, L. Hunter, B. Carpenter, R.T. Tsai, H. Dai, F. Liu, Y. Chen, C. Sun, S. Katrenko, P. Adriaans, C. Blaschke, R. Torres, M. Neves, P. Nakov, A. Divoli, J. Mata, W.J. Wilbur, Overview of bioCreative II gene mention recognition, Genome Biol. 9 (2) (2008) 1–19.
[52] M. Krallinger, F. Leitner, O. Rabal, M. Vazquez, J. Oyarzabal, A. Valencia, CHEMDNER: The drugs and chemical names extraction challenge, J. Cheminformatics 7 (2015) 1–11.