Recursively Binary Modification Model for Nested Named Entity Recognition[*]

Bing Li,[1] Shifeng Liu,[1] Yifang Sun,[1] Wei Wang,[1,2] Xiang Zhao[3]
[1]School of Computer Science and Engineering, University of New South Wales, Australia
[2]Dongguan University of Technology, China
[3]National University of Defence Technology, China
[1]{bing.li, shifeng.liu, yifang.sun, weiw}@unsw.edu.au, [3]xiangzhao@nudt.edu.cn

Abstract

Recently, there has been an increasing interest in identifying named entities with nested structures. Existing models only make independent typing decisions on the entire entity span while ignoring strong modification relations between sub-entity types. In this paper, we present a novel Recursively Binary Modification model for nested named entity recognition. Our model utilizes the modification relations among sub-entities types to infer the head component on top of a Bayesian framework and uses entity head as a strong evidence to determine the type of the entity span. The process is recursive, allowing lower-level entities to help better model those on the outer-level. To the best of our knowledge, our work is the first effort that uses modification relation in nested NER task. Extensive experiments on four benchmark datasets demonstrate that our model outperforms state-of-the-art models in nested NER tasks, and delivers competitive results with state-of-the-art models in flat NER task, without relying on any extra annotations or NLP tools.

Introduction

Named entity recognition (or more generally entity mention recognition[1]) aims at identifying text spans with regards to specific entity types such as person, organization, *etc*. Named entity recognition (NER) is a fundamental component of information extraction systems and an essential step towards many downstream NLP applications (Li et al. 2018). Significant progress has been made on NER task in the NLP community, the vast majority of which model NER task as a sequence labeling problem and employ probabilistic models (*e.g.*, CRF (Finkel, Grenager, and Manning 2005; McDonald, Crammer, and Pereira 2005; Liu et al. 2018) or Semi-CRF (Sarawagi and Cohen 2005)) on top of hand-crafted (Finkel, Grenager, and Manning 2005) or neural-based features (Lample et al. 2016).

Recently, there has been an increasing interest in identifying named entities with nested structures (*i.e.*, nested entities), where an entity is embedded within other entities, as illustrated in Fig. 1.

E1 ... their conference hosted by [The University of [Texas]LOC]ORG ...

E2 Activation of the [[[[IL-2]PROTEIN receptor]PROTEIN (IL-2R) alpha chain]PROTEIN gene]DNA, and cell proliferation.

Figure 1: Two excerpts containing nested entities.

In Fig. 1, entity `The University of Texas` of type ORG contains another entity `Texas` of type LOC. Excerpt 2 shows a DNA entity `IL-2 receptor (IL-2R) alpha chain gene` is embedded with three PROTEIN entities. Being a common linguistic phenomenon, and containing finer-grained semantic information (Katiyar and Cardie 2018), taking nested entities into consideration has been proven successful in facilitating the performance improvement of many downstream NLP applications such as relation extraction (Miwa and Bansal 2016; Liu et al. 2017), entity linking (Gupta, Singh, and Roth 2017; Miao, Qin, and Wang 2017), event extraction (Riedel and McCallum 2011; Li, Ji, and Huang 2013) and co-reference resolution (Chang, Samdani, and Roth 2013). However, traditional sequence labeling models used in flat-NER only generate one token-level label per word, which inherently do not allow nested structure, making them incapable to handle nested entities.

Existing research efforts to identify nested entities can be broadly classified into two categories: (1) transformation-based models, and (2) span-based model. Inspired by the great success of sequence labeling in detecting flat entity, transformation-based models try to transform the nested structures into flat linear structures, which can be readily solved by the sequence labeling framework. There are various transform operations, such as hyper-graph (Lu and Roth 2015; Muis and Lu 2017; Katiyar and Cardie 2018), constituency parser (Finkel and Manning 2009), shift-reduce operations (Wang et al. 2018), and stacking multiple flat-NER layers (Ju, Miwa, and Ananiadou 2018). However, they all need complex transformations and extra decoding steps, which will inevitably entail biases and errors, making the pipeline error-prone and recognition results inaccurate.

[*]B. Li and W. Wang are the co-corresponding authors.

[1]The text spans of entity mention could be named, nominal and pronominal (Florian et al. 2004).

Observing the drawback of transformation-based models, a recent work (Sohrab and Miwa 2018) proposes a simple but surprisingly effective span-based model. Instead of employing transformation to generate token-level labels, the span-based model directly generates entity type of each possible entity span on top of deep neural network generated representations. To be specific, each span is represented as the average sum of its tokens' representations, on which the model independently classifies each span into different entity types or non-entity. In this way, all nested entities are directly and independently typed without using any extra transformation step.

Despite the span-based model has achieved state-of-the-art performance in nested NER tasks up to now, its weaknesses are also obvious. Firstly, for span representation, the model simply treats all words equally, while actually, the contributions of each word could be completely different. In linguistics, the entity head is crucial to an entity's semantic type, *e.g.*, in excerpt 1, `University` plays a crucial role in determining the entity type ORG of entity mention `The University of Texas`. Simply averaging all words without distinction would dilute the significance of those important words by auxiliary words, such as `The` and `of`, and hence, yields misleading results. Secondly, the independent assumption will cause severe spurious entity structures problem (Ju, Miwa, and Ananiadou 2018) as they enumerate all spans while failing to model the dependencies among nested entities.

In order to address above issues, we propose a novel *recursively binary modification model* for nested named entity recognition. In order to weight different components within an entity to help better entity recognition, our model makes the first effort to consider modification relations among entity types to infer the head component on top of a Bayesian framework and uses entity head as strong evidence to determine entity type. The process is recursive, allowing modeling lower-level entities to help better entity recognition of outer-level, hence our model could well overcome the spurious entity structures problem. Besides, our model does not rely on any extra annotation or external knowledge resources, thus it avoids heavy training and could be easily adapted to dynamic, multilingual or domain-specific data. We evaluated our model on four benchmark datasets belonging to two different domains. Our model achieved 79.8% and 73.6% F1 scores on GENIA and ACE2005 dataset respectively.

Model

The architecture of our model is summarized in Fig. 2. From bottom to top, our model forms a three-layer layout. The first two layers are an embedding layer and a shared bidirectional LSTM (BiLSTM) layer. On top of the hidden state of the BiLSTM layer, we propose a recursively binary modification (RBM) model to identify the type of each possible entity region. We will elaborate on each layer in the following subsections.

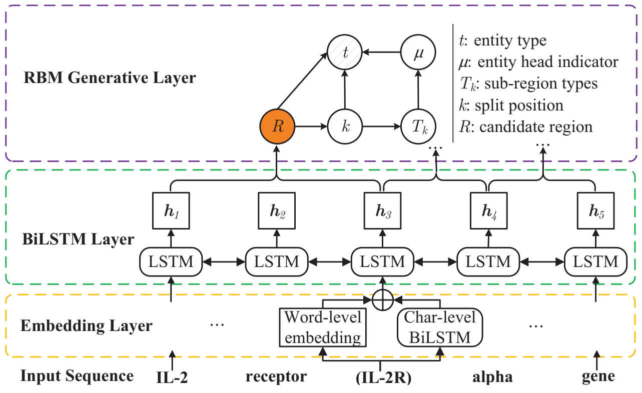


Figure 2: Architecture of the proposed model.

Embedding Layer

In the embedding layer, we concatenate a word's word-level embedding and its character-level embedding as its representation. Following the same setting with Lample *et al.* (2016), Ju *et al.* (2018) and Katiyar and Cardie (2018), word-level embeddings are initialized with pre-trained embeddings. Words in the training set but outside the pretrained embeddings are randomly initialized. Words in the testing set that are unseen in the model are mapped to a special unknow embedding; in this way, their character-level representations are highlighted.

Following the successes of Ma and Hovy (2016) and Lample *et al.* (2016) that utilized character embeddings in the NER task, we also employ character-level embeddings by concatenating forward and backward outputs of a character-level BiLSTM (char-LSTM). In our preliminary experiment, we found that the performance of char-LSTM is very close to char-CNN, similar to results in (Reimers and Gurevych 2017). Thus, in order to make a fair and side-by-side comparison to highlight the effectiveness of the proposed RBM layer, we follow the same char-LSTM settings in the embedding layer with compared methods (Ju, Miwa, and Ananiadou 2018; Wang et al. 2018; Sohrab and Miwa 2018).

BiLSTM layer

Given a word sequence $S = w_1, w_2, .., w_{|S|}$, where w_i is the i-th word in the sequence, the embedding layer transforms S into a sequence of distributed representations D_S. Then D_S is fed into a BiLSTM layer that computes the hidden states in forward $\overrightarrow{\boldsymbol{h}}= \{\overrightarrow{\boldsymbol{h}}_1, \overrightarrow{\boldsymbol{h}}_2, ..., \overrightarrow{\boldsymbol{h}}_{|S|}\}$ and backward $\overleftarrow{\boldsymbol{h}}= \{\overleftarrow{\boldsymbol{h}}_1, \overleftarrow{\boldsymbol{h}}_2, \overleftarrow{\boldsymbol{h}}_{|S|}\}$ directions. We concatenate the forward and backward as the final hidden state of the i-th word, *i.e.*, $\boldsymbol{h}_i = [\overrightarrow{\boldsymbol{h}}_i; \overleftarrow{\boldsymbol{h}}_i]$.

We use a notation $R = [i, j]$ (s.t. $1 \leq i \leq j \leq |S|$) to represent the candidate entity span region from i-th word to j-th word in a sentence S, inclusively. Based on the BiLSTM layer, the representation of region R is $\boldsymbol{h}_R = \{\boldsymbol{h}_i, \boldsymbol{h}_{i+1}, ..., \boldsymbol{h}_j\}$.

Recursively Binary Modification (RBM) Model

To weight the importances of words, instead of simply averaging the representations of tokens, a naive solution is to compute a weighted aggregation using the attention mechanism (Bahdanau, Cho, and Bengio 2014). Formally, for a given candidate entity region R and an entity type list $L = \{t_1, t_2, ..., t_{|L|}\}$, the naive solution constructs a matrix representation $\boldsymbol{v}_R = [\boldsymbol{v}_R^1, \boldsymbol{v}_R^2, ..., \boldsymbol{v}_R^{|L|}] (\boldsymbol{v}_R \in \mathbb{R}^{d\times|L|})$, in which each vector $\boldsymbol{v}_R^t \in \mathbb{R}^{d\times 1}$ is the weighted arithmetic mean over $\boldsymbol{h}_R$ related to a specific entity type t, which is defined as: $\boldsymbol{v}_R^t = \sum_{k\in R} att^t(\boldsymbol{h}_k) \cdot \boldsymbol{h}_k$, where the attention score $att^t(\boldsymbol{h}_k)$ is a softmax-normalized scalar denoting the relevance of type t and $\boldsymbol{h}_k$, which can be computed by various alignment functions (Bahdanau, Cho, and Bengio 2014), such as dot-product, feed-forward network, *etc*. Based on $\boldsymbol{v}_R$, the final entity type is the one with the highest classification score.

Unfortunately, the naive method is only able to handle nested entities sharing the same entity type; it may fail on nested entities with different types, since entity heads may correspond to different types. For example, a DNA entity `IL-2 receptor gene` is embedded within a PROTEIN entity `IL-2 receptor`. Both `receptor` and `gene` could be heads and separately present high relevance related to PROTEIN and DNA. The naive method will be unable to distinguish which one is the true head. Such a phenomenon is very common in nested entities: more than 45% of nested entities in GENIA corpus and 43% in ACE2005 corpus are embedded within another entity of a different type.

To tackle this problem, we propose a *Recursively Binary Modification model* (RBM) based on the observation that there are usually implicit modification relations among entity types, *e.g.*, a PROTEIN entity could modify a DNA or RNA entity, but not the other way around. Thus, instead of relying on "relevance" between word and type to weight the head, RBM model utilizes the modification relation among entity types to infer the head component on top of a Bayesian framework (top layer in Fig. 2). Note that our model can be recursive, allowing modeling head of lower-level entities to help better recognizing outer-level entities.

Given a candidate entity region $R = [i, j]$, RBM model predicts the output type $\hat{t}$ that obtains the maximum probability given by:

$$\hat{t} = \arg\max_{t\in L} p(t|R) = \arg\max_{t\in L} \sum_{k\in[i,j-1]} p(t, k|R), \tag{1}$$

where k is the split position that splits R into two subregions[2] $c_l = [i, k]$ and $c_r = [k + 1, j]$, one acts as the head and the other acts as the modifier.

Based on Eq. 1, we introduce a latent variable μ denoting the head among $\{c_1, c_2\}$, $p(t, k|R)$ could be probabilistically factorized as:

$$\begin{aligned} p(t, k|R) &= p(t|k, R) \cdot p(k|R) \\ &= p(k|R) \left[\sum_{\mu} p(t|\mu, k, R) \cdot p(\mu|k, R) \right], \end{aligned} \tag{2}$$

where $p(t|\mu, k, R)$ is the probability of R being an entity with label t given the head component c_μ:

$$p(t|\mu, k, R) = p(t|c_\mu). \tag{3}$$

Particularly, if $i = j$, then R is a single token region, we have:

$$p(t, k|R) = p(t|R) = \frac{\exp(\boldsymbol{v}_t \cdot \boldsymbol{h}_i)}{\sum_{\tilde{t}\in L} \exp(\boldsymbol{v}_{\tilde{t}} \cdot \boldsymbol{h}_i)}, \tag{4}$$

where $\boldsymbol{v_t}$ is the vector representation of type t.

In Eq. 2, $p(\mu, k, |R)$ denotes the probability of generating head variable μ and split position k given R, which can be factorized as:

$$p(\mu|k, R) = \sum\nolimits_{T_R\in\mathbb{T}_R} p(\mu|T_R) \cdot p(T_R|k, R), \tag{5}$$

where $\mathbb{T}_R$ represents all possible type pairs of the two subregions of R, and $T_R =< t^{c_l}, t^{c_r} > (T_R \in \mathbb{T}_R)$ is a specific type pair of components c_l and c_r. $p(\mu|T_R)$ is the probability of the μ-th component being the head component of R conditioned on the component's type pair:

$$p(\mu|T_R) = \boldsymbol{A}_{t^{\neg\mu}, t^{\mu}}, \tag{6}$$

where $\boldsymbol{A} \in [0, 1]^{|L|\times|L|}$ is a matrix of modification probabilities, in which $\boldsymbol{A}_{t_i,t_j}$ denotes the probability of a component of type t_i modifying a component of type t_j. $p(T_R|k, R)$ is the probability of generating type pair T_R, and can be computed by:

$$p(T_R|k, R) = \prod\nolimits_{q\in\{l,r\}} p(t^{c_q}|c_q). \tag{7}$$

From the Eqs. 2 and 5, latent variables μ and T_R are marginalized using the total probability rule.

For the split position k, considering that if k is right on the boundary of an identified entities or on the boundary of an existing entity in the training corpus, it would be strong evidence that we should split the entity on k. Thus, $p(k|R)$ is defined as:

$$score_k = \mathbb{I}_E(S[i, k]) + \mathbb{I}_E(S[k + 1, j]) + 1,$$

where $\mathbb{I}_E(\cdot)$ is an indicator function defined on existing entities set E, *i.e.*, $\mathbb{I}_E(S[i, k]) = 1$ if and only if $S[i, k] \in E$. Based on the score of each k, we define $p(k|R)$ as the softmax-normalized probabilities:

$$p(k|R) = \frac{\exp(score_k)}{\sum_{l\in[i,j-1]} \exp(score_l)}. \tag{8}$$

Let's take a DNA entity `IL-2 receptor gene` as an example to illustrate the recursive process of our model. Generally, to make label decision on `IL-2 receptor gene`, the computational process in the RBM will recursively break down into sub-problems by firstly identifying

[2] Please note that nested entities rarely contain crossing structures (Lu and Roth 2015), *e.g.*, it is rare for two overlapping entities w_1w_2 and w_2w_3 together embeds in the same entity $w_1w_2w_3$.

`IL-2 receptor` is a PROTEIN sub-entity (by further going down to sub-sub-problems, e.g., IL-2 + receptor) and `gene` is a DNA sub-entity. Since a DNA sub-entity could be modified by a PROTEIN sub-entity in a high probability, thus the label of `IL-2 receptor gene` depends on the label of its head part `gene`. In this manner, the label decision making of the outer-level entity could benefit from modeling lower-level sub-entities.

It is convenient to represent RBM model in matrix notations. Let $\boldsymbol{C}^k = [\boldsymbol{C}_l^k, \boldsymbol{C}_r^k]$ ($\boldsymbol{C}^k \in [0,1]^{|L| \times 2}$) be the predictive probabilities of sub-regions, in which $\boldsymbol{C}_l^k \in [0,1]^{|L| \times 1}$ and $\boldsymbol{C}_r^k \in [0,1]^{|L| \times 1}$ denote the matrices of $p(t|c_l)$ and $p(t|c_r)$ values, respectively. Let $\boldsymbol{U}^k = [p(\mu = l|k, R); p(\mu = r|k, R)]^\intercal$ ($\boldsymbol{U}^k \in [0,1]^{2 \times 1}$) is the concatenation of the two values of Eq. 5, $\boldsymbol{U}^k$ can be computed by:

$$\boldsymbol{U}^k = \left[\boldsymbol{C}_l^{k\intercal} \boldsymbol{A} \boldsymbol{C}_r^k; \boldsymbol{C}_l^{k\intercal} \boldsymbol{A}^\intercal \boldsymbol{C}_r^k\right]^\intercal. \tag{9}$$

Let $\boldsymbol{C}$ denote the matrix of $p(t|R)$ values in Eq. 1, and τ^k is a scalar denoting $p(k|R)$ value. Based on Eq. 9, we have:

$$\boldsymbol{C} = \sum\nolimits_k \tau^k \boldsymbol{C}^k \boldsymbol{U}^k. \tag{10}$$

Computational Tractability In the RBM model, the matrix $\boldsymbol{C}$ (or $p(t|k, R)$) is computed by the recursion formula $T(n) = \sum_{k=1}^{n-1}(T(k) + T(n-k))$. For an n-length region, the time complexity $T(n)$ is $\mathcal{O}(2^n)$, which is computationally intractable.

To make the computation feasible, we propose a dynamic programming strategy to avoid recomputation of overlapping sub-problems. We set up a matrix $\boldsymbol{M} \in [0,1]^{n \times n \times |L|}$, in which a cell $\boldsymbol{M}_{i,j} \in [0,1]^{|L| \times 1}$ stores a $|L|$-dimensional vector denoting the predictive probabilities $p(t|R(i,j))$. The recursion formula could be reformed as:

$$\begin{aligned} \boldsymbol{M}_{i,j} = \sum\nolimits_k & [\boldsymbol{M}_{i,k}^\intercal \boldsymbol{A} \boldsymbol{M}_{k+1,j}; \boldsymbol{M}_{i,k}^\intercal \boldsymbol{A}^\intercal \boldsymbol{M}_{k+1,j}]^\intercal \\ & \times [\boldsymbol{M}_{i,k}; \boldsymbol{M}_{k+1,j}]^\intercal \tau^k. \end{aligned} \tag{11}$$

Since matrix multiplication can be readily sped-up by GPUs, we hence assume the cost of each matrix multiplication is $\mathcal{O}(\omega)$. Using dynamic programming, for an n-length region, we only need to fill an $n \times n$ upper triangular matrix, and each computation costs $\mathcal{O}(\omega n)$, thus the complexity would be $\mathcal{O}(\omega n^3)$. Further, we could impose a maximum length constraint (Sohrab and Miwa 2018) on candidate spans, *i.e.*, only generate regions less than or equal to the length constraint. Thus the computational complexity would be $\mathcal{O}(\omega n)$, which is linear to sentence length.

Parameterization and Training The parameters of RBM model are the matrix of modification probabilities $\boldsymbol{A}$, the vector representation $\boldsymbol{v_t}$, the parameters related to hidden states $\boldsymbol{h}$ of the BiLSTM, and word-level and character-level embeddings. We use $\boldsymbol{\theta}$ to denote all the parameters collectively. $\boldsymbol{\theta}$ is trained to maximize Eq. 12 of observed entity mentions and their types in an annotated corpus.

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(t|R; \boldsymbol{\theta}). \tag{12}$$

Another issue in training is the class imbalance problem if we take all candidate entity spans into training, as overwhelming majority of them is non-entity. We employ an undersampling strategy: for a positive entity, we randomly sample m non-entities with the same span length, m also refer to the undersampling rate.

Training Objective

We used the standard cross-entropy loss as our loss function:

$$\phi = \mathcal{L}(y, \hat{t}), \tag{13}$$

where $\mathcal{L}(l, p)$ denotes cross-entropy function between l and p. $\hat{t}$ is the predicted type distribution, and y is the golden label.

Experimental Evaluation

In this section, we evaluated our model on both nested NER tasks and flat NER tasks, and demonstrate its superiority over state-of-the-art models.

Table 1: Statistics on the four datasets

Datasets	*GENIA*	*ACE2005*	*JNLPBA*	*CoNLL03*
Sentences	18546	12548	18546 (3856)	22137
Split	81:9:10	8:1:1	9:1:-	4:1:1
Outermost entities	51424	23464	-	-
Nested entities (%)	21.56	37.45	0	0
Compound entities (%)	45.41	43.49	0	0
Ave. entity length	2.90	2.28	2.18	-
Overall entities	56870	30996	51301 (8662)	34920

Evaluation Datasets

We used the following four datasets, their detailed statistics summarized in Table 1.

- **GENIA**[3] contains annotated entity mentions of 36 fine-grained entity types among 2,000 MEDLINE abstracts. We follow the same settings as Finkel and Manning (2009) and Lu and Roth (2015) by keeping only five types for evaluation, *i.e.*, DNA, RNA, PROTEIN, CELL-LINE, and CELL-TYPE. All sub-types of these five entity types are collapsed into their super-types.
- **ACE2005**[4] is a multilingual training corpus. We use its English section, which contains 7 entity types – Person (PER), Organization (ORG), Location (LOC), Geographical Entities (GPE), Vehicle (VEH), Weapon (WEA), and Facility (FAC)) among news articles. We follow the settings of Ju *et al.* (2018) by keeping files from bc, bn, cts, nw and wl and randomly split them into training, development and testing sets with the ratio 8:1:1.
- **JNLPBA**[5] contains 36 fine-grained entity types in the domain of molecular biology. Following previous research efforts (Finkel and Manning 2009; Ju, Miwa, and Ananiadou 2018), we only keep five major entity types, and randomly split 10% sentences from training data as our

[3]http://www.geniaproject.org/

[4]https://catalog.ldc.upenn.edu/LDC2006T06

[5]http://www.nactem.ac.uk/tsujii/GENIA/ERtask/report.html

development set. We use this dataset since it is widely adopted by nested NER models (Ju, Miwa, and Ananiadou 2018; Sohrab and Miwa 2018) to evaluate their ability in detecting flat entities. We follow these works for a fair comparison.

- **CoNLL03** (Tjong Kim Sang and De Meulder 2003) consists of newswire text from the Reuters RCV1 corpus tagged with four different entity types: Person (PER), Location (LOC), Organization (ORG), and Miscellaneous (MISC).

Training Details

We used pretrained embeddings on MEDLINE abstracts (Chiu et al. 2016) for *GENIA* and *JNLPBA*, and Glove[6] for *ACE2005*. Unknown words was initialized using an uniform distribution $U(-0.25, 0.25)$. All the embeddings were dynamically updated during training. We set initial modification probabilities in matrix $\boldsymbol{A}$ to $\frac{1}{|L|}$. The initial cell states and hidden states of BiLSTM were set to zero. The type representations $\boldsymbol{v}_t$ were initialized using a uniform distribution $U(0, 1)$.

The default hyper-parameter settings were: the dimension of word-level and character-level embedding was 200 and 25, respectively. The dimension of hidden state of BiLSTM was 200, and the undersampling rate m was set to 25. For optimization, we used Adam (Kinga and Adam 2015) with initial learning rate 0.001, weight-decay (L2) $1\mathrm{e}-5$, and the gradient clipping to 5; all other hyper-parameters were their default values.

Evaluation Metric

We adopted the same evaluation metric as previous works (Lu and Roth 2015; Sohrab and Miwa 2018): an entity mention is considered correct if both the mention span and the mention type are exactly correct.

Following the previous research efforts, we reported precision (P), recall (R), and micro F1 score (F1) of our best performing models on nested NER and flat NER evaluation tasks.

Nested NER Task

Compared Models For nested NER task, we compared our model RBM with seven existing models, including three state-of-the-art feature-based models (Finkel and Manning 2009; Lu and Roth 2015; Muis and Lu 2017), and four state-of-the-art deep neural models: three of them are transformation-based models (Katiyar and Cardie 2018; Wang et al. 2018; Ju, Miwa, and Ananiadou 2018), and one span-based model (Sohrab and Miwa 2018).

Results and Analysis Table 2 shows the results of our RBM model compared with state-of-the-art models. We can see that: Firstly, compared with state-of-the-art deep neural models, RBM model outperforms the best existing model by 2.7 percentage points (pts) on *GENIA*. This demonstrates our recursive binary modification model could effectively improve the accuracy in identifying the span and type of entity mention compared with the average sum representation of existing models, and account for its better performance. We also see that our model is slightly worse than Wang and Lu (2018) in *ACE2005*, but the main reason is that the score of Wang and Lu (2018)'s model is achieved using its unrestricted model that without a length constraint which would incur an $\mathcal{O}(n^3)$ computational cost, while our model only has a linear complexity $\mathcal{O}(n)$. For its restricted version (with $\mathcal{O}(n^2)$ complexity), the performance is below ours (72.8% v.s. 73.6%). Thus, we believe that the overall performance of our model is still better than Wang and Lu (2018)'s model.

Secondly, compared with state-of-the-art feature-based models, our model outperforms the best feature-based model (Muis and Lu 2017) by 9 pts and 10.5 pts on *GENIA* and *ACE2005*, respectively. This again demonstrates that deep neural models have huge advantages over traditional feature-based models.

Thirdly, among the two datasets, our model gains more improvement on *GENIA* than *ACE2005*. The main reason is that, modification relations are commonly stronger in the biomedical domain (*GENIA*) than in news articles domain (*ACE2005*). For example, entities in biomedical domain often present strong modification patterns such as a PROTEIN or RNA entity could modify a DNA entity, whereas for PERSON or LOCATION entities of the news domain, such relations are commonly weaker.

Another observation is that, our model perform with better precision on *ACE2005* dataset but with better recall on *GENIA*. We believe that it is mainly caused by different parameter settings. Obviously, there is a precision/recall tradeoff w.r.t undersampling rate m (as shown in Fig. 4). If we set $m = 15$, the results on *ACE2005* would be 75.3%(P) 71.6%(R) and 73.4%(F1), perform better in all aspects P (+1.1%), R (+1.2%) and F1 (+0.4%) than best baseline.

Table 3 shows the performances of our model on different entity levels compared with the best existing model Sohrab and Miwa (2018). Our model outperforms the best existing model in both flat, nested, single-token, and multi-token entities. This demonstrates our model could well handle different entity levels.

Table 4 shows categorical performances of the RBM model on *GENIA* dataset, also compared with the best existing model Sohrab and Miwa (2018). Besides the best performing category PROTEIN, which has the largest number of training instances, our model performs equally well on all categories. Compared with Sohrab and Miwa's model, our model beats it in all categories except slightly worse in CELL-TYPE. Another observation is, our model performs significantly better in the categories which commonly has strong modification relations among inner-entities, such as DNA and CELL-LINE, which demonstrates our RBM model could well model the modification relation and help with better entity recognition.

Taken *GENIA* dataset as the example, we show modification probabilities (parameter $\boldsymbol{A}$) learned by RBM and visualize its strength as a heatmap in Fig. 3. We can see that the probabilities of any entity type modifying a non-entity are quite small, since non-entities could rarely be the head.

[6]https://nlp.stanford.edu/projects/glove/

Table 2: Comparisons of our model with state-of-the-art models on nested NER tasks, the best results are marked in bold font. Sohrab and Miwa did not report results on *ACE2005*, so the scores marked with $*$ are estimated using our self-implemented version.

Model	*GENIA*			*ACE2005*		
	P (%)	R (%)	F1 (%)	P (%)	R (%)	F1 (%)
Finkel and Manning (2009)	75.4	65.9	70.3	-	-	-
Lu and Roth (2015)	72.5	65.2	68.7	70.0	56.9	62.8
Muis and Lu (2017)	75.4	66.8	70.8	69.1	58.1	63.1
Katiyar and Cardie (2018)	76.7	71.1	73.8	70.6	70.4	70.5
Wang *et al.* (2018)	-	-	73.9	-	-	73.0
Ju *et al.* (2018)	78.5	71.3	74.7	74.2	70.3	72.2
Wang and Lu (2018)	77.0	73.3	75.1	76.8	72.3	**74.5**
Sohrab and Miwa (2018)	93.2	64.0	77.1	82.0*	63.7*	71.7*
RBM	82.5	77.4	**79.8**	79.7	68.4	73.6

Table 3: Performance comparison of the RBM model and the model by Sohrab and Miwa v.s. different entity-level on *GENIA* dataset.

Entity-level	RBM			Sohrab & Miwa
	P (%)	R (%)	F1 (%)	F1 (%)
Single-token	84.2	79.6	**81.8**	69.9
Multi-token	80.9	75.3	**78.0**	77.9
Flat	81.5	77.8	**79.6**	79.3
Nested	98.0	72.2	**83.1**	72.7
All entities	82.5	77.4	**79.8**	77.1

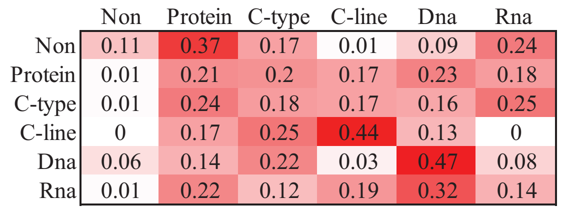
Table 4: Categorical performance comparison of the RBM model and the model by Sohrab and Miwa on *GENIA* dataset.

Categories	RBM			Sohrab & Miwa
	P (%)	R (%)	F1 (%)	F1 (%)
DNA	78.4	71.8	**75.0**	71.8
CELL-LINE	75.8	69.9	**72.7**	67.9
CELL-TYPE	81.7	72.4	76.8	**78.1**
PROTEIN	84.8	80.8	**82.7**	80.8
RNA	70.7	79.5	**74.8**	72.4
Overall	82.5	77.4	**79.8**	77.1

Some common patterns are correctly learned, such as a PROTEIN or RNA entity could modify a DNA entity, but not the other way round. The probabilities also reveal the fact that, some strong types likes CELL-LINE and DNA rarely modify other types, but they are of high probabilities modifying themselves. From Fig. 3, we can conclude that the modification probabilities leaned by our model are meaningful and in accord with common sense.

Ablation Study To evaluate the contribution of modification, pretrained embeddings and character level embeddings components to final results, we conducted an ablation study by ablating a specific component from the full model once per test. The results are shown in Table 5.

We can see that ablation on modification probabilities incurs roughly 2 pts reduction in terms of F1 score, which demonstrates modification component is necessary to improve the performance. Also, ablations on character-level



	Non	Protein	C-type	C-line	Dna	Rna
Non	0.11	0.37	0.17	0.01	0.09	0.24
Protein	0.01	0.21	0.2	0.17	0.23	0.18
C-type	0.01	0.24	0.18	0.17	0.16	0.25
C-line	0	0.17	0.25	0.44	0.13	0
Dna	0.06	0.14	0.22	0.03	0.47	0.08
Rna	0.01	0.22	0.12	0.19	0.32	0.14

Figure 3: The modification probabilities $\boldsymbol{A}$ learned on *GENIA* dataset, in which a cell $\boldsymbol{A}_{i,j}$ donates the probability of an entity of type i that modifies an entity of type j.

embeddings and pretrained embeddings lead to big performance gaps on both datasets. From above we can conclude that these components contribute significantly to the effectiveness of our model.

Sensitivity Analysis of the Undersampling Rate The undersampling rate m is an important hyper-parameter in handling the class-imbalance problem. In order to test its efforts on precision, recall, and the F1 score, and thereby offer an empirical way to tune its setting, we conducted a sensitivity analysis on *GENIA* and *ACE2005* datasets. The results are shown in Fig. 4.

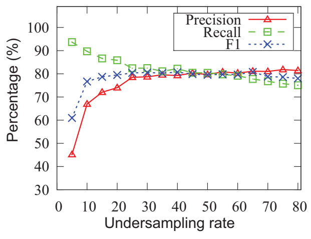
From Fig. 4, we can see that despite *GENIA* and *ACE2005* are from different domains, the tendencies are similar. To be specific, a small undersampling rate rewards high recall but low precision, whereas a larger undersampling rate rewards high precision but low recall. With the increasing of the undersampling rate, F1 scores quickly rise to a flat peak region and then have a slight drop down. In terms of F1, the most appropriate undersampling rate setting is supposed to be among 25 - 60. Further, considering the extra computational cost brought by a larger m, a smaller rate ($m = 25$) is preferable.

Flat NER Task

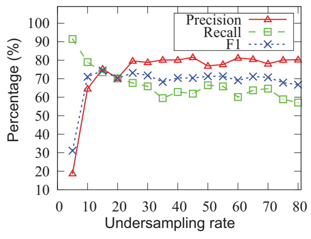
We also tested our model RBM on the flat NER task to demonstrate our model could not only outperform existing models in nested NER tasks but also perform well in identifying flat entities. We tested our model on the *JNLPBA*

Table 5: Ablation study results compared with the full RBM model.

Model	*GENIA*			*ACE2005*		
	P (%)	R (%)	F1 (%)	P (%)	R (%)	F1 (%)
RBM	82.5	**77.4**	**79.8**	**79.7**	68.4	**73.6**
- modification probabilities	**83.1**	72.9	77.7	77.2	**68.9**	72.8
- pretrained embeddings	78.2	71.0	74.4	79.3	65.1	71.5
- character-level embeddings	74.7	59.3	66.1	73.6	60.8	66.6



(a) GENIA.



(b) ACE2005.

Figure 4: Precision, recall, and F1 score v.s. various undersampling rate tested on Dev set.

dataset, which is a commonly used evaluation dataset containing only flat entities of the biomedical domain. Moreover, we also tested on general domain *CoNLL03* dataset. We compared with six state-of-the-art models for reference. The results are listed in Table 6.

Table 6: Results of the flat NER task.

Model	*JNLPBA*	*CoNLL03*
	F1 (%)	F1 (%)
Gridach (2017)	75.87	-
Ju *et al.* (2018)	75.55	-
Sohrab and Miwa (2018)	**78.4**	-
Wang and Lu (2018)	-	90.20
Lample *et al.* (2016)	-	90.94
Ma and Hovy (2016)	-	91.21
Yang *et al.* (2017)	-	**91.26**
RBM	77.6	91.17

From Table 6, we can see that, on *JNLPBA* dataset, our model beats Gridach (2017) and Ju *et al.* (2018) by nearly 2 pts. Albeit our model performs slightly worse than the best existing model Sohrab and Miwa (2018) in terms of F1 score, but the difference is only 0.8 pt. On general domain *CoNLL03* dataset, our model obtains an F1 score of 91.17%, which is competitive with state-of-the-art flat NER models, the difference with the best model is only 0.09 pt, which is fully acceptable.

We wish to point out that identifying flat entities is not the primary purpose of our model, the performance on the flat NER task is more than expected.

Related Works

Nested NER task has gained an increasing research interest in recent years. Existing nested NER models can be broadly classified into transformation-based models and span-based model according to the methodologies they adopted.

The transformation-based models are directly derived from the sequence labeling framework used in the flat NER task. Conventional sequence labeling framework only generates one output label per word, thus it is incapable to handle nested structure. Therefore, transformation-based models try to transform the nested NER problem into a standard sequence labeling problem using various transformations. A common transformation is hyper-graph (Lu and Roth 2015; Muis and Lu 2017; Katiyar and Cardie 2018). A hyper-graph can be viewed as multiple parallel linear chain CRFs, which allows generating multiple labels on each word, and adapts to nested structures. Another transformation is dynamically stacking multiple flat NER layers (Ju, Miwa, and Ananiadou 2018). Each layer is used to identify entities on a specific nest level, and the stacking stops if no entities can be predicted. Wang *et al.* (2018) proposed a shift-reduce based model to transform the nested NER task into an action sequence labeling problem, and the final nested entities can be fetched by a decoding process. Finkel and Manning (2009) represented nested entities as nodes in a constituency parsing tree, and the types are decided by a CRF-based approach. The transformation-based models can be directly built upon existing state-of-the-art flat NER models to detect nested entities. However, the complex transformations and extra decoding steps will inevitably entail biases and errors in nested NER task.

Sohrab and Miwa (2018) adopted a different methodology and proposed a span-based model. Instead of tagging each word by sequence labeling, the model directly generated entity types on each candidate span based on features generated by deep neural networks. This model achieves state-of-the-art results in both nested and flat NER tasks. However, they treat all tokens equally, which will entail noisy information. Besides, its independent assumption makes it ignoring dependencies between nested entities. By contrast, our model utilizes a recursive binary modification framework to weight different components and captures the dependencies to help better nested entity recognition.

Conclusion

In this paper, we explore how to recursively utilize modification relations among entity types to better recognize nested entities. Our model utilizes the modification relations among sub-entities types to infer the head component on top of a Bayesian framework and uses entity head as strong evidence to determine the type of the entity span. Our model achieved 79.8% and 73.6% F1 scores on GENIA and

ACE2005 dataset respectively, without relying on any extra annotations or NLP tools.

For future work, we will investigate the use of external information (*e.g.*, dependency relation) for further performance improvement.

Acknowledgements. This work is supported by ARC DPs 170103710 and 180103411, D2DCRC DC25002 and DC25003, NSFC under grant No. 61872446 and PNSF of Hunan under grant No. 2019JJ20024. The Titan V used for this research was donated by the NVIDIA Corporation.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Chang, K.-W.; Samdani, R.; and Roth, D. 2013. A constrained latent variable model for coreference resolution. In *EMNLP*, 601–612.

Chiu, B.; Crichton, G.; Korhonen, A.; and Pyysalo, S. 2016. How to train good word embeddings for biomedical nlp. In *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*, 166–174.

Finkel, J. R., and Manning, C. D. 2009. Nested named entity recognition. In *EMNLP*, 141–150. ACL.

Finkel, J. R.; Grenager, T.; and Manning, C. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *ACL*, 363–370. ACL.

Florian, R.; Hassan, H.; Ittycheriah, A.; Jing, H.; Kambhatla, N.; Luo, X.; Nicolov, N.; and Roukos, S. 2004. A statistical model for multilingual entity detection and tracking. In *HLT-NAACL*.

Gridach, M. 2017. Character-level neural network for biomedical named entity recognition. *Journal of biomedical informatics* 70:85–91.

Gupta, N.; Singh, S.; and Roth, D. 2017. Entity linking via joint encoding of types, descriptions, and context. In *EMNLP*, 2681–2690.

Ju, M.; Miwa, M.; and Ananiadou, S. 2018. A neural layered model for nested named entity recognition. In *NAACL-HLT*, volume 1, 1446–1459.

Katiyar, A., and Cardie, C. 2018. Nested named entity recognition revisited. In *NAACL-HLT*, 861–871. ACL.

Kinga, D., and Adam, J. B. 2015. A method for stochastic optimization. In *ICLR*, volume 5.

Lample, G.; Ballesteros, M.; Subramanian, S.; Kawakami, K.; and Dyer, C. 2016. Neural architectures for named entity recognition. In *NAACL-HLT*, 260–270.

Li, J.; Sun, A.; Han, J.; and Li, C. 2018. A survey on deep learning for named entity recognition. *CoRR* abs/1812.09449.

Li, Q.; Ji, H.; and Huang, L. 2013. Joint event extraction via structured prediction with global features. In *ACL*, volume 1, 73–82.

Liu, L.; Ren, X.; Zhu, Q.; Zhi, S.; Gui, H.; Ji, H.; and Han, J. 2017. Heterogeneous supervision for relation extraction: A representation learning approach. In *EMNLP*, 46–56.

Liu, S.; Sun, Y.; Wang, W.; and Zhou, X. 2018. A crf-based stacking model with meta-features for named entity recognition. In *PAKDD*, 54–66. Springer.

Lu, W., and Roth, D. 2015. Joint mention extraction and classification with mention hypergraphs. In *EMNLP*, 857–867.

Ma, X., and Hovy, E. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *ACL*, volume 1, 1064–1074.

McDonald, R.; Crammer, K.; and Pereira, F. 2005. Flexible text segmentation with structured multilabel classification. In *HLT-EMNLP*, 987–994. ACL.

Miao, Y.; Qin, J.; and Wang, W. 2017. Graph summarization for entity relatedness visualization. In *SIGIR*, 1161–1164. ACM.

Miwa, M., and Bansal, M. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *ACL*, volume 1, 1105–1116.

Muis, A. O., and Lu, W. 2017. Labeling gaps between words: Recognizing overlapping mentions with mention separators. *EMNLP*.

Reimers, N., and Gurevych, I. 2017. Reporting score distributions makes a difference: Performance study of lstm-networks for sequence tagging. In *EMNLP*, 338–348.

Riedel, S., and McCallum, A. 2011. Fast and robust joint models for biomedical event extraction. In *EMNLP*, 1–12. ACL.

Sarawagi, S., and Cohen, W. W. 2005. Semi-markov conditional random fields for information extraction. In *NIPS*, 1185–1192.

Sohrab, M. G., and Miwa, M. 2018. Deep exhaustive model for nested named entity recognition. In *EMNLP*, 2843–2849.

Tjong Kim Sang, E. F., and De Meulder, F. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *HLT-NAACL*, CONLL '03, 142–147. Stroudsburg, PA, USA: ACL.

Wang, B., and Lu, W. 2018. Neural segmental hypergraphs for overlapping mention recognition. In *EMNLP*, 204–214.

Wang, B.; Lu, W.; Wang, Y.; and Jin, H. 2018. A neural transition-based model for nested mention recognition. In *EMNLP*, 1011–1017.

Yang, Z.; Salakhutdinov, R.; and Cohen, W. W. 2017. Transfer learning for sequence tagging with hierarchical recurrent networks. *ICLR*.