Chinese NER Using Lattice LSTM

Yue Zhang*, Jie Yang*   * equal contribution

Singapore University of Technology and Design

Overview

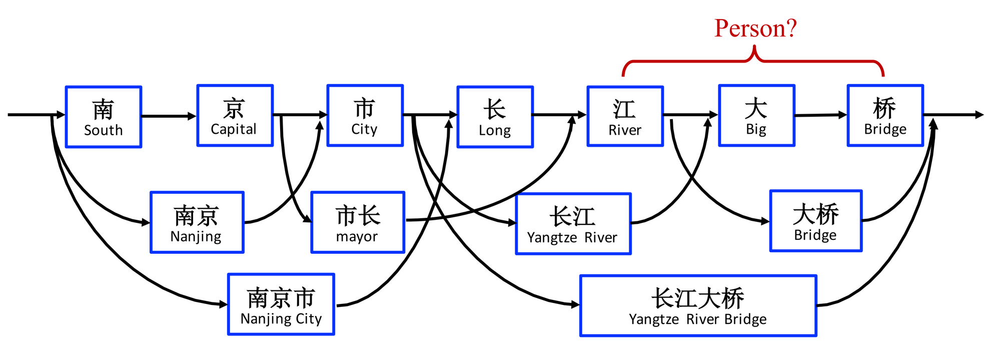
❖ **Named Entity Recognition:** locate and classify text segments into pre-defined categories such as person, location etc.

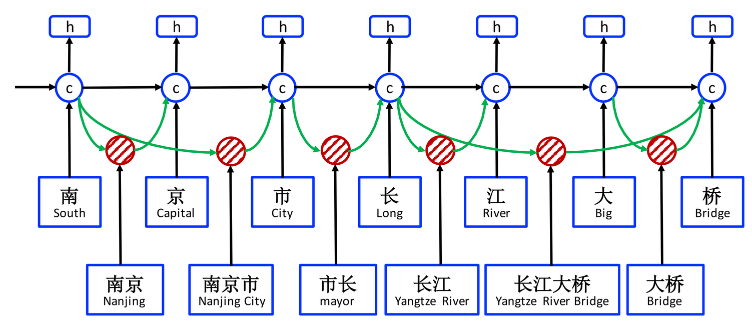
我和 [美国]$_{\text{Location}}$的 [华莱士]$_{\text{Person}}$先生聊天.

I talked with Mr. [Wallace]$_{\text{Person}}$ from [United States]$_{\text{Location}}$.

❖ **Chinese NER:** character information and word information

- **Character-based models:** take the character sequence as input, then label each character.
  Hard to utilize word sequence information.
- **Word-based models:** text are segmented as word sequence and label each word.
  Suffers from segmentation error propagation.
- **Lattice models:**
  Character-based model with word lattice shortcut connection.
  Interaction of both word and character sequence.





Models

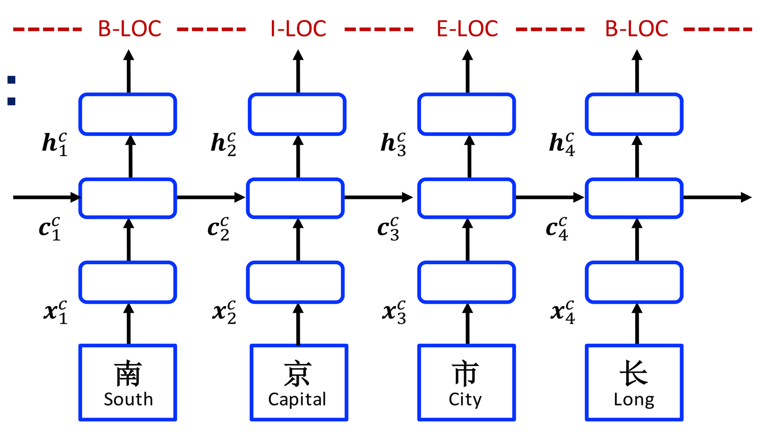
❖ **LSTM: (coupled)**

- **Char-based and word based models have the same structure:**

$$\begin{bmatrix} \mathbf{i}_j \\ \mathbf{o}_j \\ \widetilde{\mathbf{c}}_j \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ tanh \end{bmatrix} \left(\mathbf{W}^\top \begin{bmatrix} \mathbf{x}_j \\ \mathbf{h}_{j-1} \end{bmatrix} + \mathbf{b} \right)$$

$$\mathbf{c}_j = (1 - \mathbf{i}_j) \odot \mathbf{c}_{j-1} + \mathbf{i}_j \odot \widetilde{\mathbf{c}}_j$$

$$\mathbf{h}_j = \mathbf{o}_j \odot tanh(\mathbf{c}_j)$$

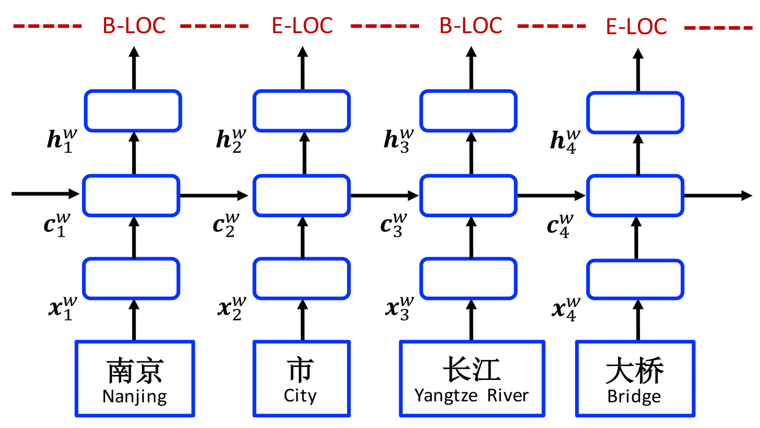


❖ **Lattice LSTM:**

- **Lattice path calculation:**

$$\begin{bmatrix} \mathbf{i}^w_{b,e} \\ \mathbf{f}^w_{b,e} \\ \widetilde{\mathbf{c}}^w_{b,e} \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ tanh \end{bmatrix} \left(\mathbf{W}^{w\top} \begin{bmatrix} \mathbf{x}^w_{b,e} \\ \mathbf{h}^c_b \end{bmatrix} + \mathbf{b}^w \right)$$

$$\mathbf{c}^w_{b,e} = \mathbf{f}^w_{b,e} \odot \mathbf{c}^c_b + \mathbf{i}^w_{b,e} \odot \widetilde{\mathbf{c}}^w_{b,e}$$

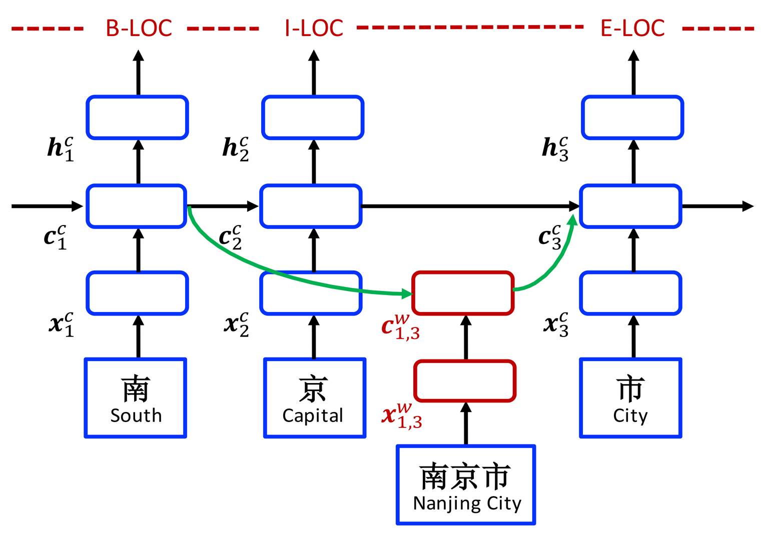


- **Lattice & Character calculation:**

$$\begin{bmatrix} \mathbf{i}_j \\ \mathbf{o}_j \\ \widetilde{\mathbf{c}}_j \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ tanh \end{bmatrix} \left(\mathbf{W}^\top \begin{bmatrix} \mathbf{x}_j \\ \mathbf{h}_{j-1} \end{bmatrix} + \mathbf{b} \right)$$

$$\mathbf{c}_j = \sum_{b \in \{b' | w^d_{b',j} \in \mathbb{D}\}} \boldsymbol{\alpha}_{b,j} \odot \mathbf{c}^w_{b,j} + \boldsymbol{\alpha}_j \odot \widetilde{\mathbf{c}}_j$$

$$\mathbf{h}_j = \mathbf{o}_j \odot tanh(\mathbf{c}_j)$$



$$\boldsymbol{\alpha}_{b,j} = \frac{exp(\mathbf{i}_{b,j})}{exp(\mathbf{i}_j) + \sum_{b' \in \{b'' | w^d_{b'',j} \in \mathbb{D}\}} exp(\mathbf{i}_{b',j})}$$

$$\boldsymbol{\alpha}_j = \frac{exp(\mathbf{i}_j)}{exp(\mathbf{i}_j) + \sum_{b' \in \{b'' | w^d_{b'',j} \in \mathbb{D}\}} exp(\mathbf{i}_{b',j})}$$

use multiple normalized gates to control the contributions of different lattice paths.

Experiments

❖ **Datasets:** four Chinese NER datasets

- **OntoNotes 4:** news domain, with 4entity types.
- **MSRA**: news domain, with 3 entity types.
- **Weibo NER:** social media NER corpus.
- **Resume NER**: manual annotated, with 8 entity types.

❖ **Segmentation:**.

- **Segmentor:** SOTA word segmentor in Yang et al. ACL 2017
- **Lexicon/Word embeddings:** auto-segmented Chinese Gigaword with the above segmentor and trained with word2vec.

Baselines: LSTM+CRF

❖ **Word baselines:** word-based LSTM+CRF models

- **+char LSTM:** with extra char LSTM to represent word.
- **+char+bichar+LSTM**: extra char+bichar LSTM to represent word.
- **+char CNN:** with extra char CNN to represent word.
- **+char+bichar+CNN**: extra char+bichar CNN to represent word.

❖ **Character baselines:** character-based LSTM+CRF models.

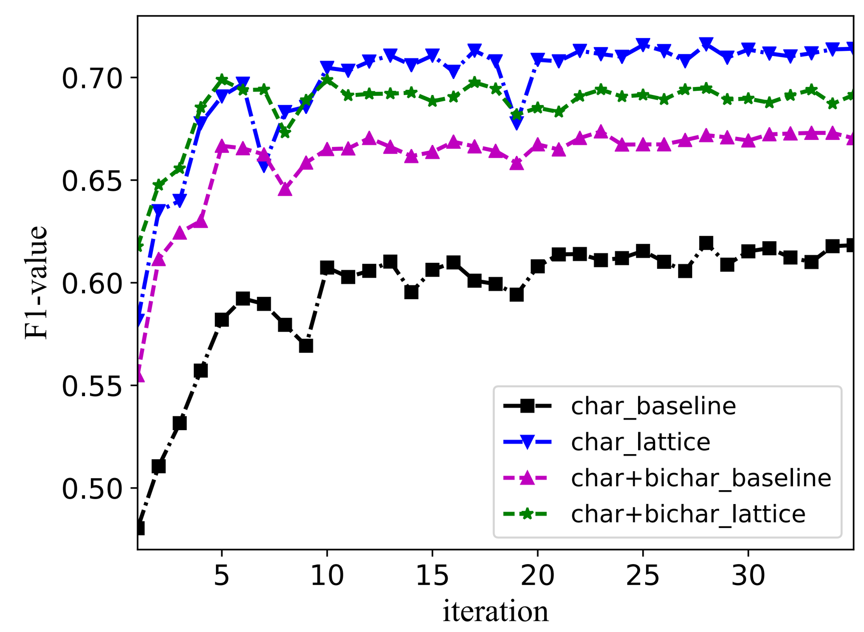
- **+softword:** auto-segmentation results as neural features.
- **+bichar:** with extra char bigram embeddings.
- **+bichar+softword:** with both extra bigram and softword features.

Results

❖ **Dev Results:**

- **Char-based NER:** char+bichar+softword gives the best result.
- **Word-based NER**: word+char+bichar LSTM gives the best result.
- **Lattice NER:** significantly improves the accuracy compared with both char-based and word-based baselines.
- **Char vs Lattice**: char bigram information is useful in char-based baseline, while it does not improve the accuracy of lattice LSTM.

Input	Models	P	R	F1
Auto seg	Word baseline	73.20	57.05	64.12
	+char LSTM	71.98	65.41	68.54
	+char LSTM′	71.08	65.83	68.35
	+char+bichar LSTM	72.63	67.60	70.03
	+char CNN	73.06	66.29	69.51
	+char+bichar CNN	72.01	65.50	68.60
No seg	Char baseline	67.12	58.42	62.47
	+softword	69.30	62.47	65.71
	+bichar	71.67	64.02	67.63
	+bichar+softword	72.64	66.89	69.64
	Lattice	**74.64**	**68.83**	**71.62**



❖ **Final Results:**

- **OntoNotes:** lattice LSTM significantly outperforms all baselines.
- **MSRA**: previous state-of-the-are achieves 90.9% F1-value, our lattice LSTM significantly boosts the result as 93.18%.
- **Weibo & Resume:** lattice LSTM also has significant improvement on small datasets.

OntoNotes

Input	Models	P	R	F1
Gold seg	Yang et al. (2016)	65.59	71.84	68.57
	Yang et al. (2016)*†	72.98	**80.15**	**76.40**
	Che et al. (2013)*	77.71	72.51	75.02
	Wang et al. (2013)*	76.43	72.32	74.32
	Word baseline	76.66	63.60	69.52
	+char+bichar LSTM	**78.62**	73.13	75.77
Auto seg	Word baseline	72.84	59.72	65.63
	+char+bichar LSTM	73.36	70.12	71.70
No seg	Char baseline	68.79	60.35	64.30
	+bichar+softword	74.36	69.43	71.81
	Lattice	**76.35**	**71.56**	**73.88**

MSRA

Models	P	R	F1
Chen et al. (2006a)	91.22	81.71	86.20
Zhang et al. (2006)*	92.20	90.18	91.18
Zhou et al. (2013)	91.86	88.75	90.28
Lu et al. (2016)	–	–	87.94
Dong et al. (2016)	91.28	90.62	90.95
Word baseline	90.57	83.06	86.65
+char+bichar LSTM	91.05	89.53	90.28
Char baseline	90.74	86.96	88.81
+bichar+softword	92.97	90.80	91.87
Lattice	**93.57**	**92.79**	**93.18**

Weibo

Models	NE	NM	Overall
Peng and Dredze (2015)	51.96	61.05	56.05
Peng and Dredze (2016)*	**55.28**	**62.97**	**58.99**
He and Sun (2017a)	50.60	59.32	54.82
He and Sun (2017b)*	54.50	62.17	58.23
Word baseline	36.02	59.38	47.33
+char+bichar LSTM	43.40	60.30	52.33
Char baseline	46.11	55.29	52.77
+bichar+softword	50.55	60.11	56.75
Lattice	**53.04**	**62.25**	**58.79**

Resume

Models	P	R	F1
Word baseline	93.72	93.44	93.58
+char+bichar LSTM	94.07	94.42	94.24
Char baseline	93.66	93.31	93.48
+bichar+softword	94.53	**94.29**	94.41
Lattice	**94.81**	94.11	**94.46**

Analysis

❖ **F1 with Sentence Length:**

- **Char baseline:** is not sensitive with the sentence length.
- **Word baseline**: works worse with the increase of sentence length, since the segmentor accuracy is worse in long sentences.
- **Lattice LSTM:** In general, it gives better performance in all sentence length. It also suffers the accuracy deduction in long sentences, which can result from an exponentially increasing number of word combination in the lattice.