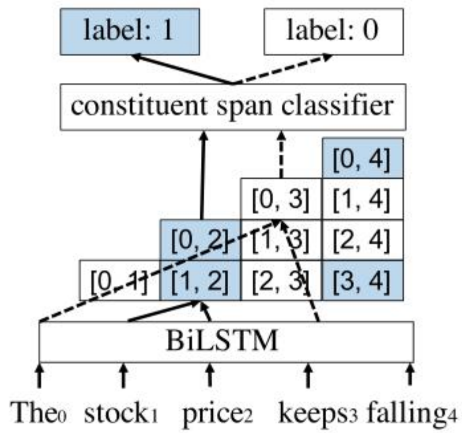
Two Local Models for Neural Constituent Parsing

WeiYang
weiyang@godweiyang.com
https://godweiyang.com

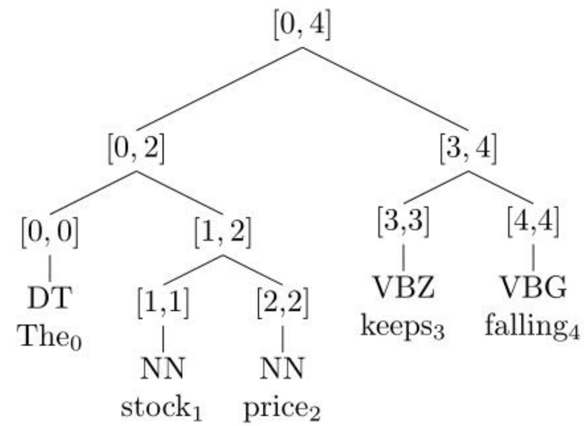
Introduction

- **Non-local** features have been shown crucial for **statistical parsing**.
- However, **local** models have been shown to give highly competitive accuracies for **neural parsing**.
- What extent the encoding power can be leveraged for constituent parsing?
- We investigate two local neural models which make local decisions to **constituent spans** and **CFG rules**.

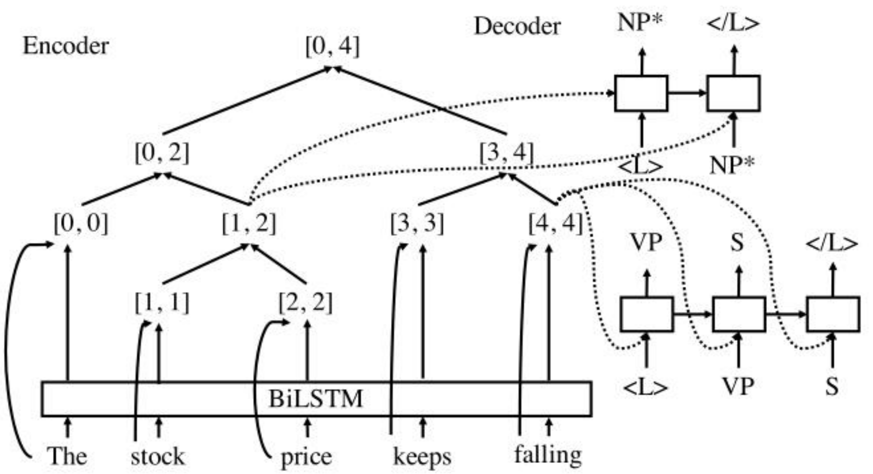
Model



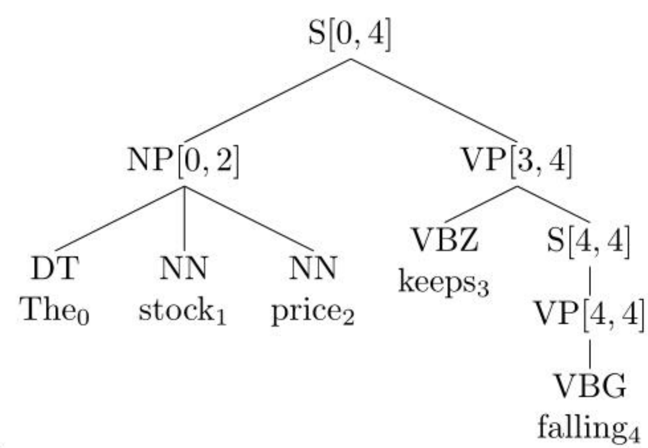
(a) BiLSTM based constituent span classifier.



(b) unlabeled binarized parse tree.



(c) Label generator for two example spans. **NP*** is an intermediate constituent label.



(d) Final example parse tree.

Unlabeled Parser

- **Span model**
  Identifies the probability of an arbitrary span being a constituent span.
- **Rule model**
  Considers the probability $P([i,j] \rightarrow [i,k][k+1,j]|S)$ for the production rule that the span $[i,j]$ is composed by two children spans $[i,k]$ and $[k+1,j]$.

Span Model

- Span representation

$$v[i,j] = [f_{i+1}; r_i; f_{j+1}; r_j]$$

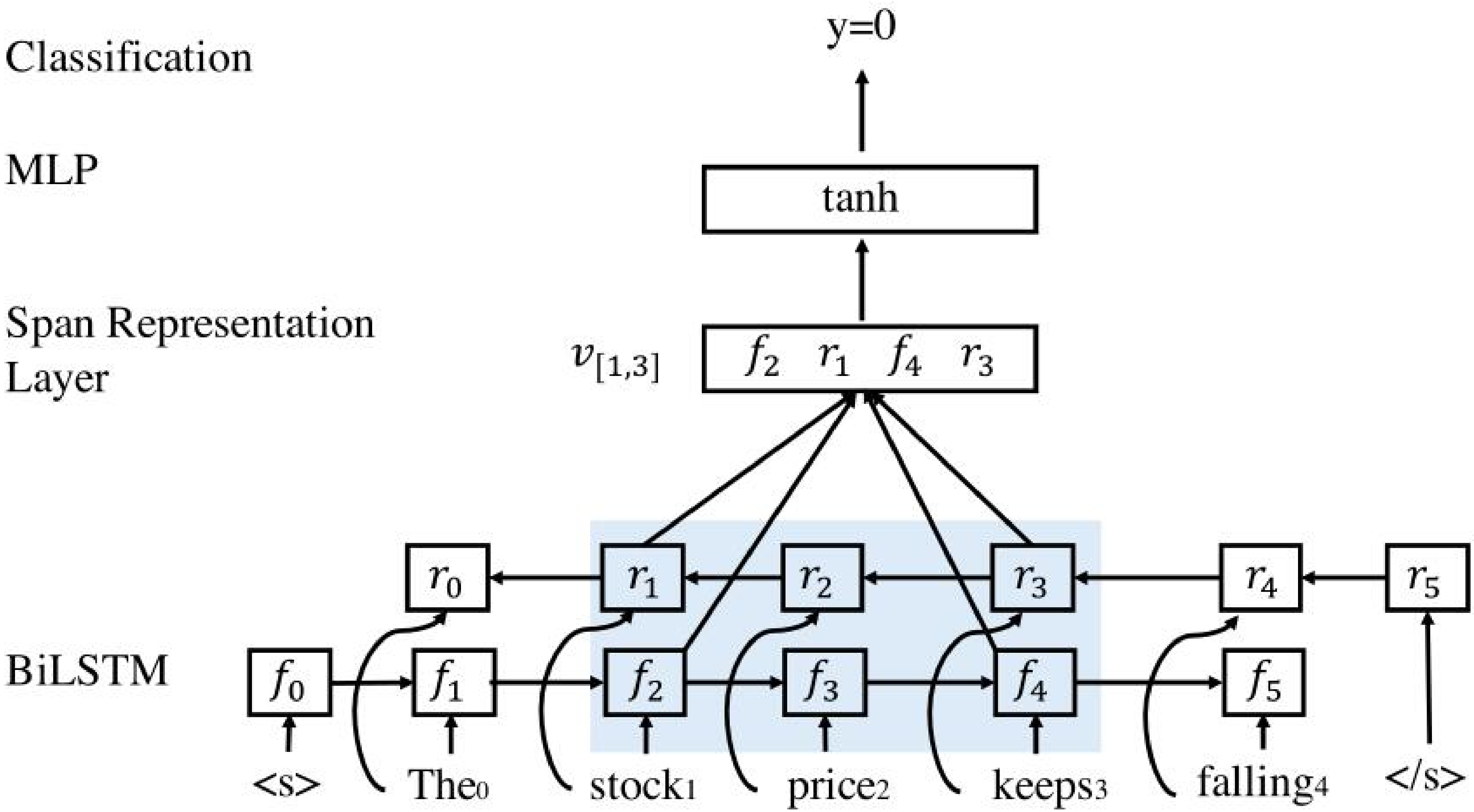
- Probability distribution

$$P(Y_{[i,j]}|S,\Theta) = softmax(f(v[i,j]))$$

- Training objective

$$\mathcal{L}_{binary} = -\sum_{[i,j]\in T_{ub}} \log P(Y_{[i,j]} = 1) - \sum_{[i,j]\notin T_{ub}} \log P(Y_{[i,j]} = 0)$$

Span Model



(a) Span model.

Neural CKY Algorithm

The unlabeled production probability for the rule $r : [i, j] \rightarrow [i, k][k + 1, j]$ given by the binary classification model is

$$P(r|S, \Theta) = P(Y_{[i,k]} = 1|S, \Theta)P(Y_{[k+1,j]} = 1|S, \Theta)$$

Multi-class Span Classification Model

- Probability distribution

$$P(Y_{[i,j]} = c|S, \Theta) = softmax(g(v[i,j]))_{[c]}$$

- Training objective

$$\mathcal{L}_{multi} = -\sum_{[i,j] \in T_{ub}} \sum_{c \in GEN[i,j], c \neq </L>} \log P(Y_{[i,j]} = c)$$
$$- \sum_{[i,j] \notin T_{ub}} \log P(Y_{[i,j]} =< /L >)$$

Neural CKY Algorithm

Transform the multi-class probability distribution into a binary probability distribution by using

$$P(Y_{[i,j]} = 1|S, \Theta) = \sum_{c, c \neq </L>} P(Y_{[i,j]} = c|S, \Theta)$$

$$P(Y_{[i,j]} = 0|S, \Theta) = P(Y_{[i,j]} =< /L > |S, \Theta)$$

Rule Model

- **Span representation**

$$
\begin{aligned}
s[i,j] &= [f_{j+1} - f_i; r_i - r_{j+1}] \\
sr[i,j] &= [s[0,i-1]; s[i,j]; s[j+1,n-1]] \\
r[i,j] &= \phi(W_r^M sr[i,j] + b_r^M)
\end{aligned}
$$

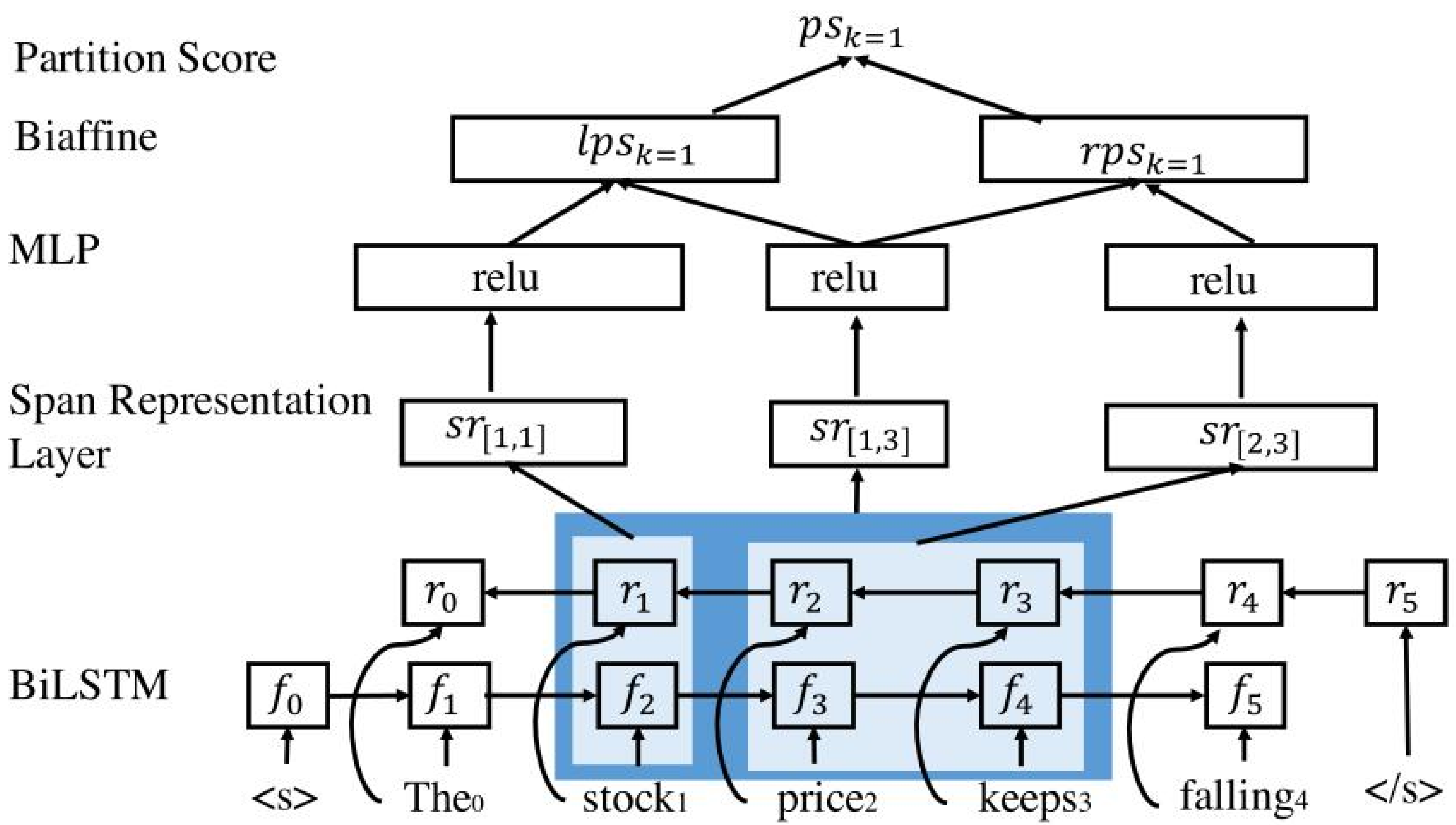
- **Linear model**

$$
ps_k = w_{ll,k}^T r[i,k] + w_{lr,k}^T r[k+1,j] + b_{ll,k}
$$

- **Biaffine model**

$$
\begin{aligned}
lps_k &= (r[i,j] \oplus 1)^T W_{pl} (r[i,k] \oplus 1) \\
rps_k &= (r[i,j] \oplus 1)^T W_{pr} (r[k+1,j] \oplus 1) \\
ps_k &= lps_k + rps_k
\end{aligned}
$$

Rule Model



(b) Rule model.

Rule Model

- **Unlabeled production probability**

$$P(r|S,\Theta) = \frac{\exp(ps_k)}{\sum_{k'=i}^{j-1} \exp(ps_{k'})}$$

- **Training objective**

$$\mathcal{L}_{rule} = -\sum_{r \in T_{ub}} \log P(r|S,\Theta)$$

Label Generator

- **Lexicalized tree-LSTM encoder**
  Use lexicalized tree-LSTM encoder to get the representation $h[i, j]$ of span $[i, j]$.
- **Label decoder**
  Suppose the constituent label chain for the span $[i, j]$ is $(L_p^0, L_p^1, \dots, L_p^m)$, the probability distribution of generating the label at time step z is

$$
\begin{aligned}
&P(L_p^z | T_{ub}, L_p^{z<m}) \\
&= softmax(g(h[i, j], E_{label}(L_p^{z-1}), d_{z-1}))
\end{aligned}
$$

Label Generator

- Training objective

$$\mathcal{L}_{label}[i,j] = -\sum_{z=0}^{m} \log P(L_p^z | T_{ub}, L_p^{z<m})$$

$$\mathcal{L}_{label} = \sum_{[i,j] \in T_{ub}} \mathcal{L}_{label}[i,j]$$

Joint Training

- **Training objective**

$$\mathcal{L}_{total} = \mathcal{L}_{parser} + \mathcal{L}_{label} + \frac{\lambda}{2}\|\Theta\|^2$$

Experiments

Model	SpanVec	LP	LR	LF
BinarySpan	$\mathbf{v}[i,j]$	**92.16**	**92.19**	**92.17**
	$\mathbf{sr}[i,j]$	91.90	91.70	91.80
BiaffineRule	$\mathbf{v}[i,j]$	91.79	91.67	91.73
	$\mathbf{sr}[i,j]$	**92.49**	**92.23**	**92.36**

Table 2: Span representation methods.

Experiments

Model	English			Chinese		
	LP	LR	LF	LP	LR	LF
BinarySpan	92.16	92.19	92.17	91.31	90.48	90.89
MultiSpan	92.47	**92.41**	**92.44**	**91.69**	90.91	**91.30**
LinearRule	92.03	92.03	92.03	91.03	89.19	90.10
BiaffineRule	**92.49**	92.23	92.36	91.31	**91.28**	91.29

Table 3: Main development results.

Experiments

Parser	LR	LP	LF	Parser	LR	LP	LF
Zhu et al. (2013) (S)	91.1	91.5	91.3	Charniak (2000)	89.5	89.9	89.5
McClosky et al. (2006) (S)	92.1	92.5	92.3	Collins (2003)	88.1	88.3	88.2
Choe and Charniak (2016) (S,R,E)			93.8	Sagae and Lavie (2006)	87.8	88.1	87.9
Durrett and Klein (2015) (S)			91.1	Petrov and Klein (2007)	90.1	90.2	90.1
Vinyals et al. (2015) (S, E)			92.8	Carreras et al. (2008)	90.7	91.4	91.1
Charniak and Johnson (2005) (S, R)	91.2	91.8	91.5	Zhu et al. (2013)	90.2	90.7	90.4
Huang (2008b) (R)			91.7	Watanabe and Sumita (2015)			90.7
Huang and Harper (2009) (ST)	91.1	91.6	91.3	Fernández-González and Martins (2015)	89.9	90.4	90.2
Huang et al. (2010) (ST)	92.7	92.2	92.5	Cross and Huang (2016b)	90.5	92.1	91.3
Shindo et al. (2012) (E)			92.4	Kuncoro et al. (2017)			91.2
Socher et al. (2013) (R)			90.4	Liu and Zhang (2017b)	91.3	92.1	91.7
Dyer et al. (2016) (R)			93.3	Stern et al. (2017) top-down	90.4	93.2	91.8
Kuncoro et al. (2017) (R)			93.6	**BinarySpan**	91.9	92.2	92.1
Liu and Zhang (2017a) (R)			94.2	**MultiSpan**	92.2	92.5	**92.4**
Fried et al. (2017) (ES)			**94.7**	**BiaffineRule**	92.0	92.6	92.3

Table 4: Results on the PTB test set. *S* denotes parsers using auto parsed trees. *E*, *R* and *ST* denote ensembling, reranking and self-training systems, respectively.