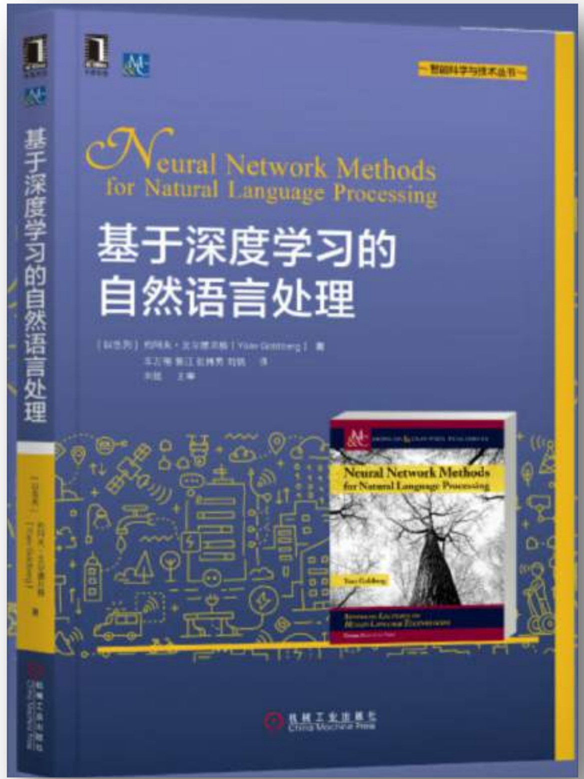
Deep Learning for Structured Prediction in Natural Language Processing

Wanxiang Che（车万翔）
Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology
2018-12-22

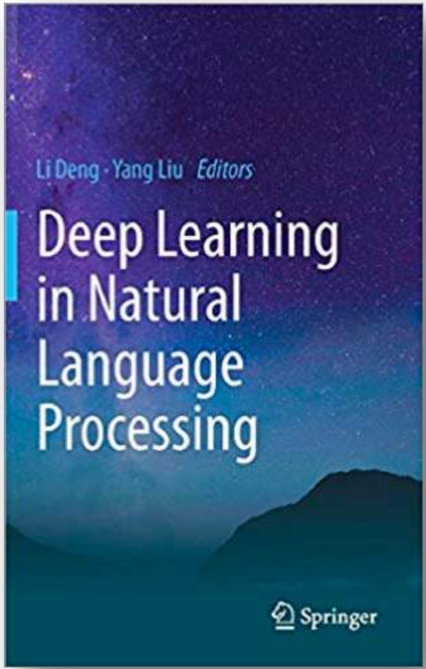
Reference Book

- 基于深度学习的自然语言处理
  - Neural Network Methods for Natural Language Processing
  - 约阿夫·戈尔德贝格（Yoav Goldberg）著
  - 车万翔、郭江、张伟男、刘铭（译）
  - 机械工业出版社出版
  - 2018年5月



Reference Book

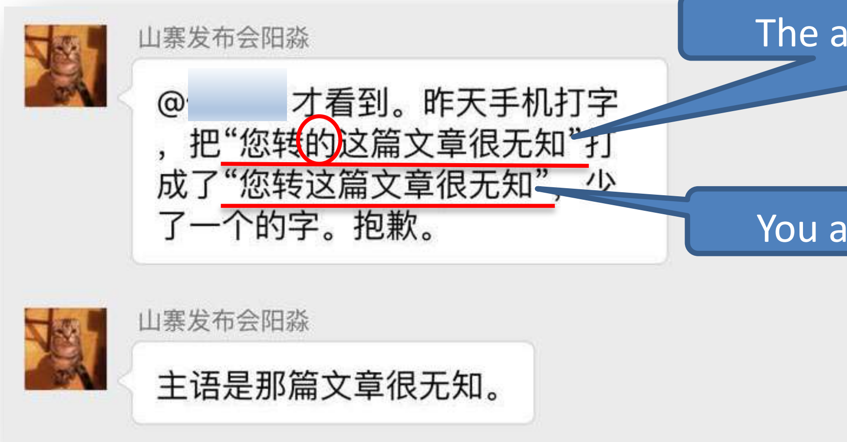
- Deep Learning in Natural Language Processing
  - Editors: Li Deng and Yang Liu
  - Springer, 2018
- **Chapter 4: Deep Learning in Lexical Analysis and Parsing**
  - Wanxiang Che and Yue Zhang



Part 1: Structured Prediction

Part 1.1: Fundamental NLP Tasks

Why Do We Need Structures?

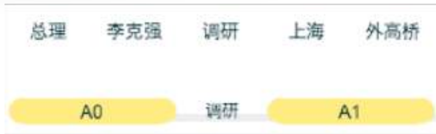


The article you retweeted is ignorant.

You are ignorant to retweet the article.

- Parsing proposes the (syntactic or semantic) relations between words
- These relations are important for many applications

Fundamental NLP Pipeline



总理/n [李克强 **人名**] 调研/v [上海 外高桥 **地名**]

总理/n 李克强/nh 调研/v 上海/ns 外高桥/ns

总理 李克**强 调**研 上海 外高桥

总理李克强调研上海外高桥

Word Segmentation

- Words are fundamental semantic units
- Chinese has no obvious word boundaries
- Word segmentation
  - Split Chinese character sequence into words
- Ambiguities in word segmentation
  - E.g. 严守一把手机关了
    - 严守一/ 把/ 手机/ 关/ 了
    - 严守/ 一把手/ 机关/ 了
    - 严守/ 一把/ 手机/ 关/ 了
    - 严守一/ 把手/ 机关/ 了
    - ……

Part-of-speech (POS) Tagging

- A POS is a category of words which have **similar grammatical properties**
  - E.g. noun, verb, adjective
- POS tagging
  - Marking up a word in a text as a particular POS based on both its definition and its **context**
- Ambiguities in POS Tagging
  - Time flies like an arrow.
  - 制服了敌人 vs. 穿着制服

Named Entity Recognition (NER)

- Named Entities
  - Persons, locations, organizations, expressions of times, quantities, monetary values, percentages, etc.
- Locating and classifying named entities in text into pre-defined categories
- Ambiguities in NER

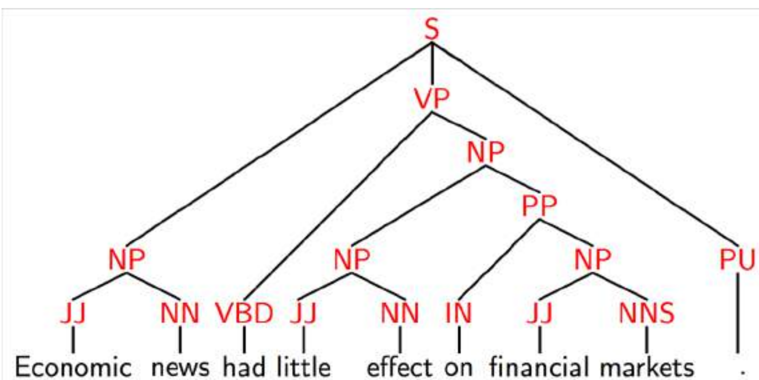
Kerry to visit Jordan, Israel Palestinian peace on agenda.

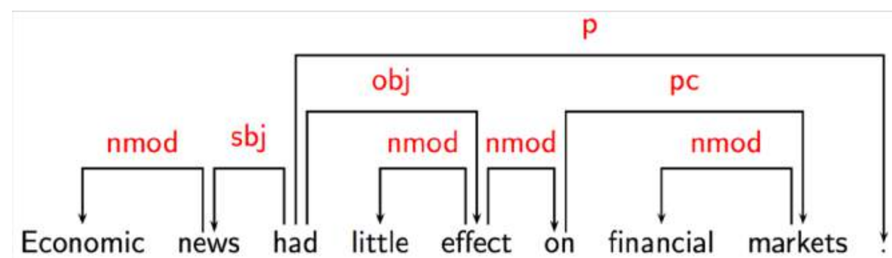
Jordan → 人名(乔丹)？

Jordan → 地名(约旦)？ √

Syntactic Parsing

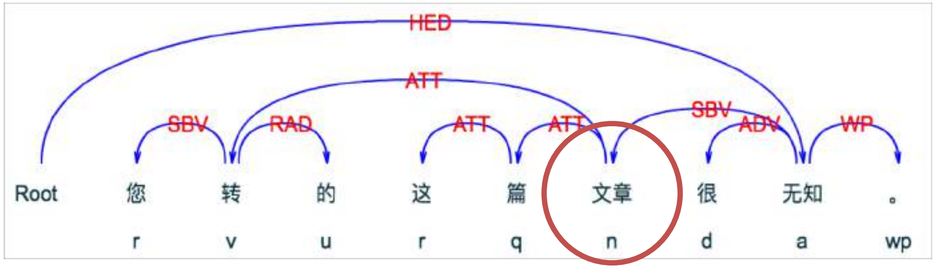
- Analyzing a natural language string conforming to the rules of a formal grammar, emphasizing subject, predicate, object, etc.
  - Constituency and Dependency Parsing

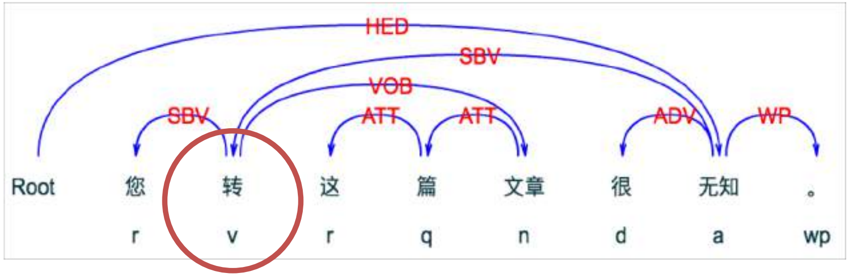




Dependency Parsing

- A dependency tree is a tree structure composed of the input words and satisfies a few constraints:
  - Single-head
  - Connected
  - Acyclic





http://ltp.ai/demo.html

Semantic Role Labeling

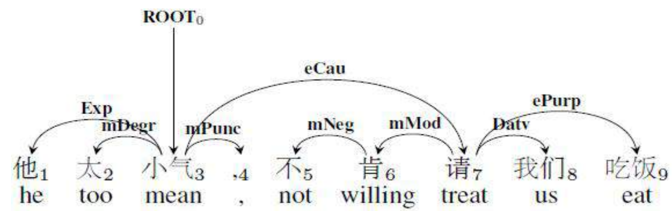
- Recognizing predicates and corresponding arguments

Example from (Yih & Toutanova, 2006)

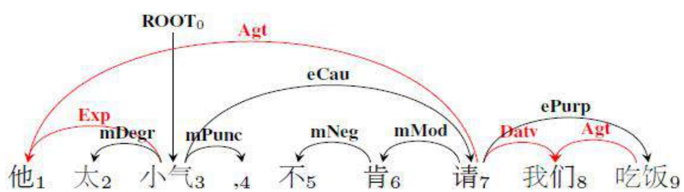
Semantic Role Labeling

- Answer "Who did what to whom when and where"
  - Question Answering
    - Yesterday time , Mary buyer bought a shirt bought thing from Tom seller
    - Whom buyer did Tom seller sell a shirt bought thing to, yesterday time
  - Information Extraction
  - ……

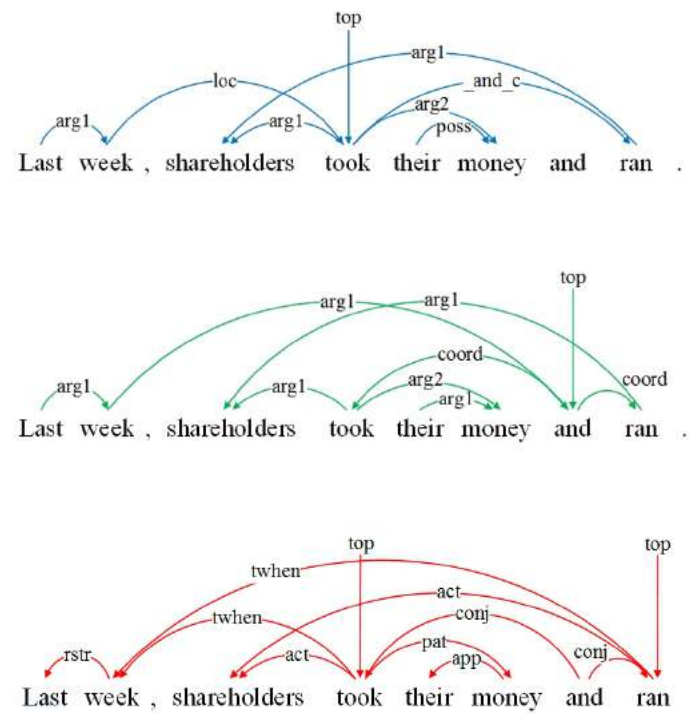
Semantic Dependency Graph



SemEval 2012 Task 5 : Chinese Semantic Dependency (Tree)



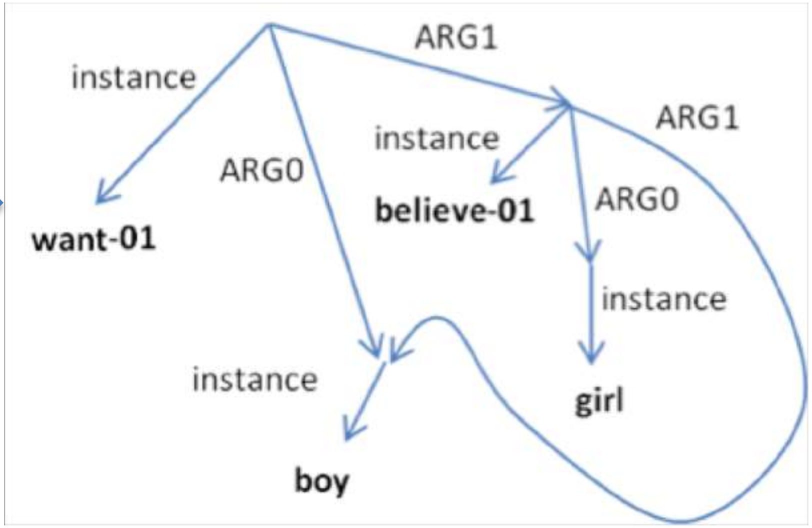
SemEval 2016 Task 9 : Chinese Semantic Dependency (Graph)



SemEval 2015 Task 18: Broad-Coverage Semantic Dependency (Graph)

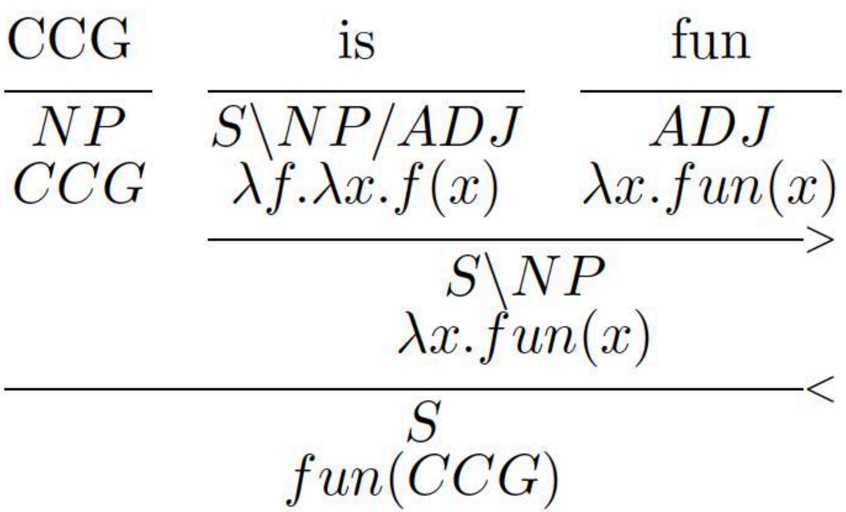
Abstract Meaning Representation (AMR)

The boy wants the girl to believe him.
The boy wants to be believed by the girl.
The boy has a desire to be believed by the girl.
The boy's desire is for the girl to believe him.
The boy is desirous of the girl believing him.



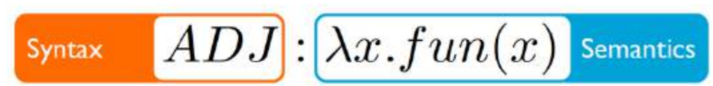
http://www.isi.edu/~ulf/amr/help/amr-guidelines.pdf

Combinatory Categorial Grammars (CCG)

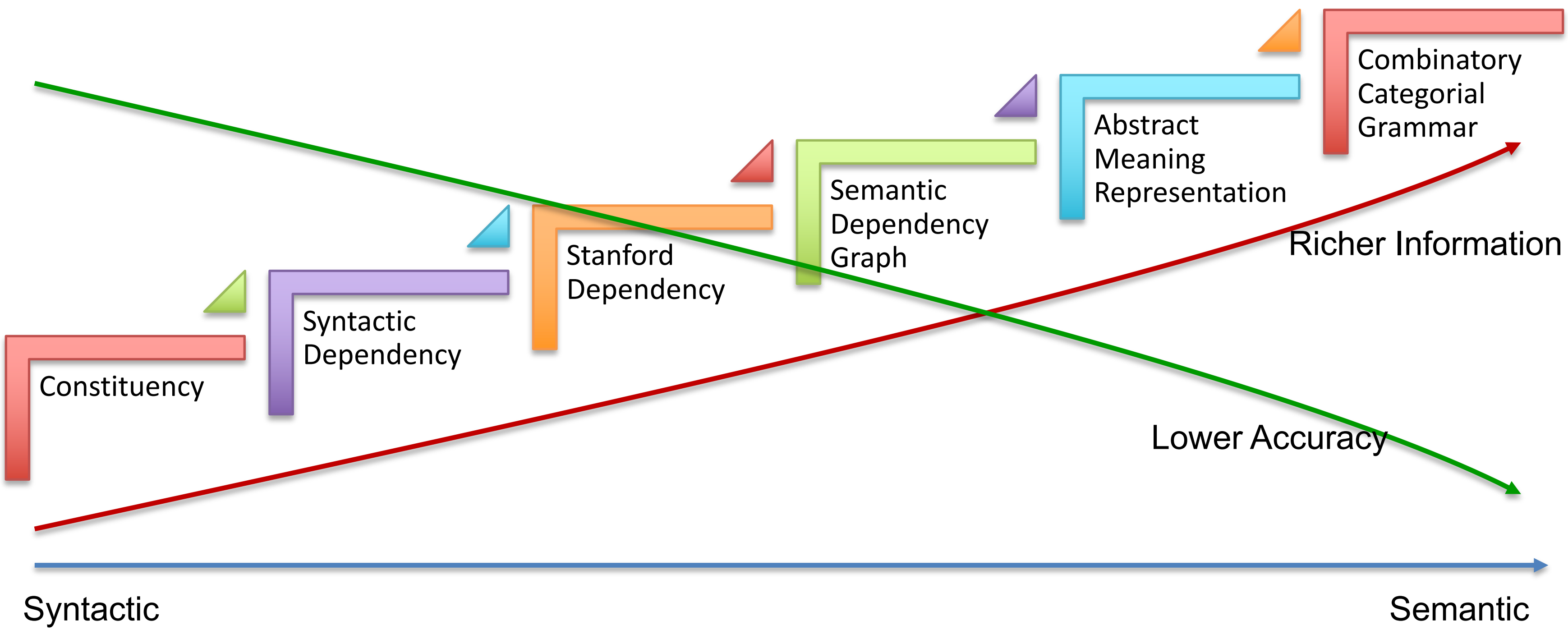


- CCG Lexical Entries
  - Pair words and phrases with meaning by a CCG category

- CCG Categories
  - Basic building block
  - Capture syntactic and semantic information jointly



Grammar



Constituency
Syntactic Dependency
Stanford Dependency
Semantic Dependency Graph
Abstract Meaning Representation
Combinatory Categorial Grammar
Richer Information
Lower Accuracy
Syntactic
Semantic

Part 1.2: Structured Prediction

Structured Prediction

- Predicting structured objects, rather than scalar discrete or real values
- Outputs are **influenced each other**
- Three categories
  - Sequence segmentation
  - Sequence labeling / Tagging
  - Parsing

Sequence Segmentation

- Break a sequence into contiguous parts
- For example: Word Segmentation
  - Input
    - 严守一把手机关了
  - Output
    - 严守一/ 把/ 手机/ 关/ 了/
- More examples:
  - Sentence segmentation (a post-processing stage for speech transcription)
  - Paragraph segmentation

Sequence Labeling/Tagging

- Given an input sequence, produce a **label sequence** of equal length
- Each label is drawn from a small finite set
- Labels are **influenced each other**
- For example: POS tagging
  - Input
    - Profits soared at Boeing Co., easily topping forecasts on Wall Street, …
  - Output
    - Profits/N soared/V at/P Boeing/N Co./N ,/, easily/ADV …

NER

- Input
  - Profits soared at Boeing Co., easily topping forecasts on Wall Street, …
- Output
  - Profits soared at [Boeing Co. $_{ORG}$], easily topping forecasts on [Wall Street $_{LOC}$], …
- Alternative Output (Tagging)
  - Profits/O soared/O at/O Boeing/B-ORG Co./I-ORG ,/O easily/O topping/O forecasts/O on/O Wall/B-LOC Street/I-LOC ,/O …
- Where
  - B: Begin of entity XXX; I: Inside of entity XXX; O: Others

Word Segmentation

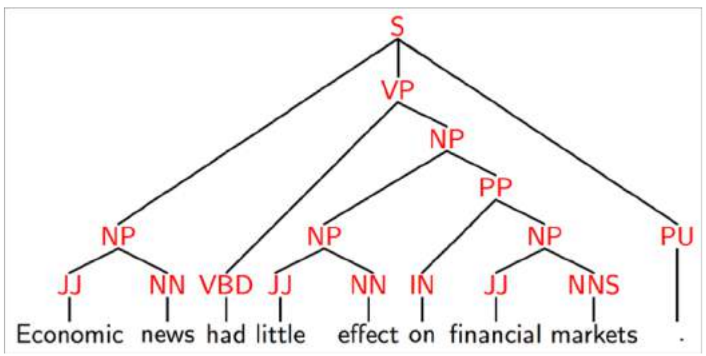
- Input
  - 严守一把手机关了
- Output
  - 严守一/ 把/ 手机/ 关/ 了/
- Alternative Output (Tagging)
  - 严/B 守/I 一/I 把/B 手/B 机/I 关/B 了/B
- Where
  - B: Begin of a word; I: Inside of a word

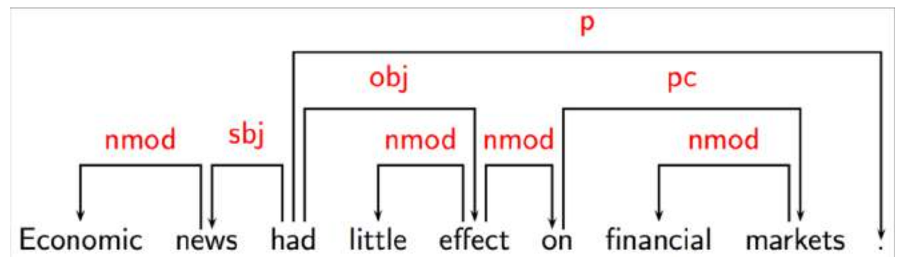
Semantic Role Labeling

- Input
  - Yesterday, Mary bought a shirt from Tom
- Output
  - [Yesterday $_{time}$], [Mary $_{buyer}$] bought/pred [a shirt $_{bought\ thing}$] from [Tom $_{seller}$]
- Alternative Output (Tagging)
  - Yesterday/B-time ,/O Mary/B-buyer bought/pred a/B-bought thing shirt/I-bought thing from/O Tom/B-seller
- Where
  - B: Begin of an arg; I: Inside of an arg; O: Others

Parsing Algorithms

- All kinds of algorithms converting sentences to tree or graph structures
  - Constituency and **Dependency Parsing**





Part 1: Summary

- NLP Tasks
  - Word segmentation, POS tagging, named entity recognition
  - Constituent/dependency parsing
  - Semantic Role Labeling, Semantic (graph) dependency parsing
  - Abstract Meaning Representation (AMR)
  - Combinatory Categorial Grammars (CCG)
- Structured Prediction
  - Sequence segmentation
  - Sequence labeling / Tagging
  - Parsing

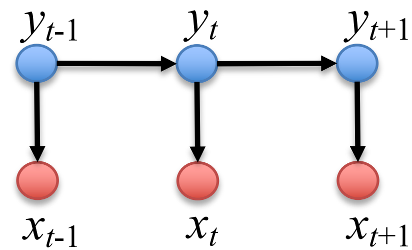
Part 2: Graph-based Methods

Part 2.1: Graph-based Sequence Labeling

Traditional Sequence Labeling Models

HMM

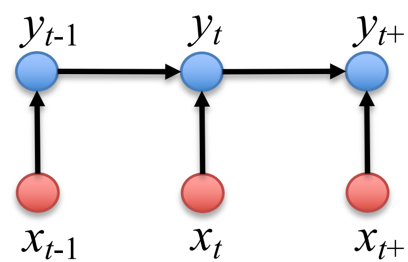
$$P(y_{[1:n]}, x_{[1:n]}) \propto \prod_{t=1}^{n} P(y_t|y_{t-1}) P(x_t|y_t)$$



MEMM

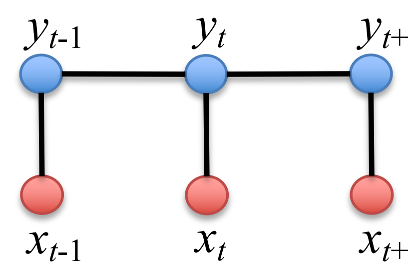
$$P(y_{[1:n]}|x_{[1:n]}) \propto \prod_{t=1}^{n} P(y_t|y_{t-1}, x_t)$$

$$\propto \prod_{t=1}^{n} \frac{1}{Z_{y_{t-1},x_t}} \exp\left(\begin{matrix}\sum_j \lambda_j f_j(y_t, y_{t-1}) \\ + \sum_k \mu_k g_k(y_t, x_t)\end{matrix}\right)$$



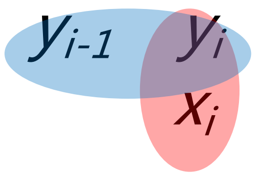
CRF

$$P(y_{[1:n]}|x_{[1:n]}) \propto \frac{1}{Z_{y_{[1:n]}}} \prod_{t=1}^{n} \exp\left(\begin{matrix}\sum_j \lambda_j f_j(y_t, y_{t-1}) \\ + \sum_k \mu_k g_k(y_t, x_t)\end{matrix}\right)$$



Features of POS Tagging with CRF

- Assume only two feature templates
  - tag bigrams
  - word/tag pairs



$$f_{100} = \begin{cases} 1 \text{ if } <y_{i-1}, y_i> \; = \; <n, v> \\ \quad\quad 0 \text{ otherwise} \end{cases}$$

$$g_{101} = \begin{cases} 1 \text{ if } x_i \text{ is ended with “ing” and } y_i \; = \; v \\ \quad\quad 0 \text{ otherwise} \end{cases}$$

CRF Decoding

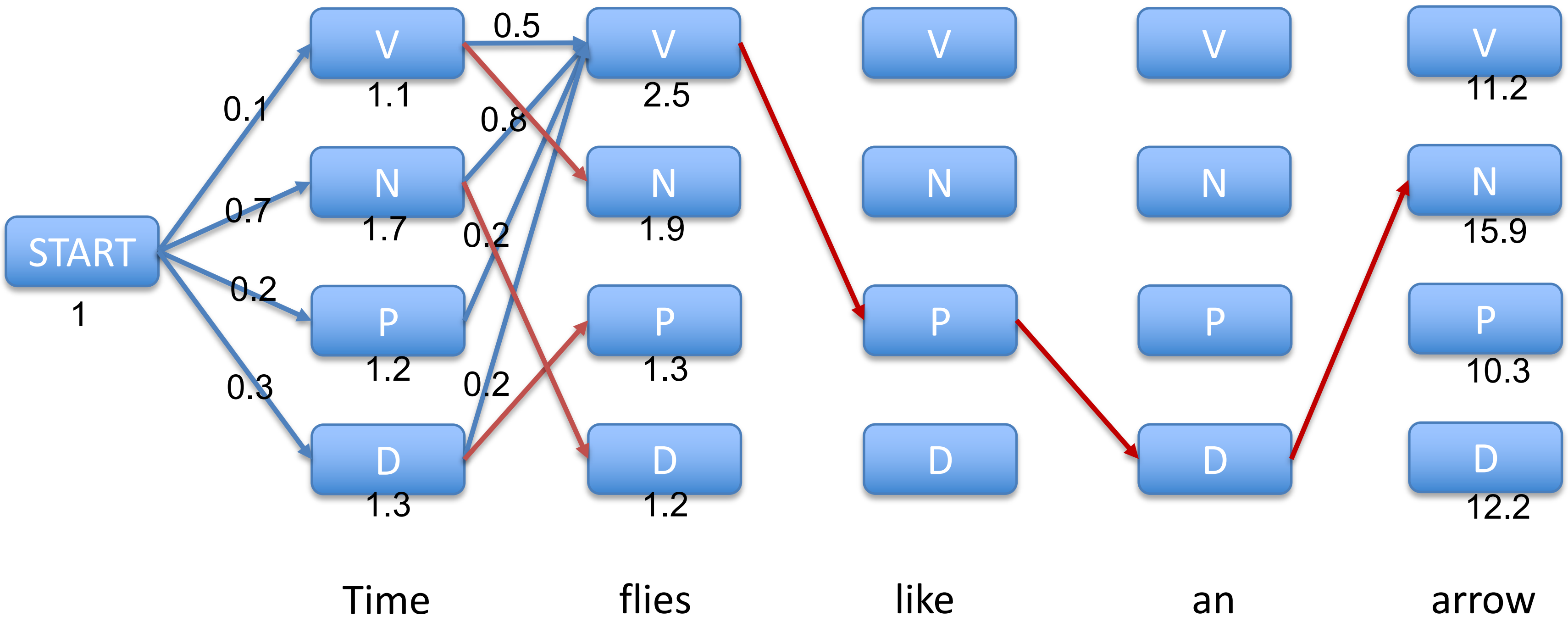
$$\arg\max_{y_{[1:n]} \in \mathrm{GEN}(x_{[1:n]})} \sum_{i=1}^{n} \mathbf{w} \cdot \mathbf{f}(x_{[1:n]}, y_i, y_{i-1})$$

where $\mathrm{GEN}(x_{[1:n]})$ is all possible tag sequences

- Dynamic Programming Algorithm
  - Viterbi Algorithm

Viterbi Algorithm

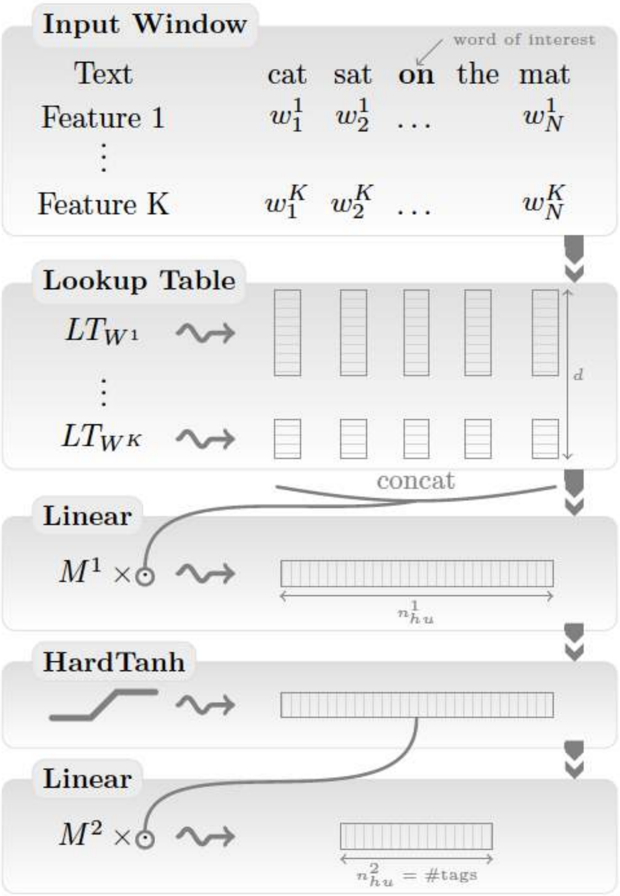
- Define a dynamic programming table
  - $\pi(i, y)$ = maximum score of a tag sequence ending in tag y at position i
- Recursive definition: $\pi(i, y) = \max_t \left(\pi(i-1, t) + \mathbf{w} \cdot \mathbf{f}\left(x_{[1:n]}, y, t\right)\right)$



Deep Learning for Sequence Labeling

- Window Approach
  - Tag **one word** at a time
  - Feed a **fixed-size** window of text around **each word** to tag
- Features
  - Words, POS tags, Suffix, Cascading, …

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. J. Mach. Learn. Res. 12, 2493-2537.



Sentence-Level Log-Likelihood

- Considering dependencies between tags in a sentence
- Conditional likelihood by normalizing all possible paths (CRF)
- Sentence score for one tag path

$$\log p([y]_1^T \mid [x]_1^T, \tilde{\theta}) = s([x]_1^T, [y]_1^T, \tilde{\theta}) - \underset{\forall [j]_1^T}{\text{logadd}}\, s([x]_1^T, [j]_1^T, \tilde{\theta})$$

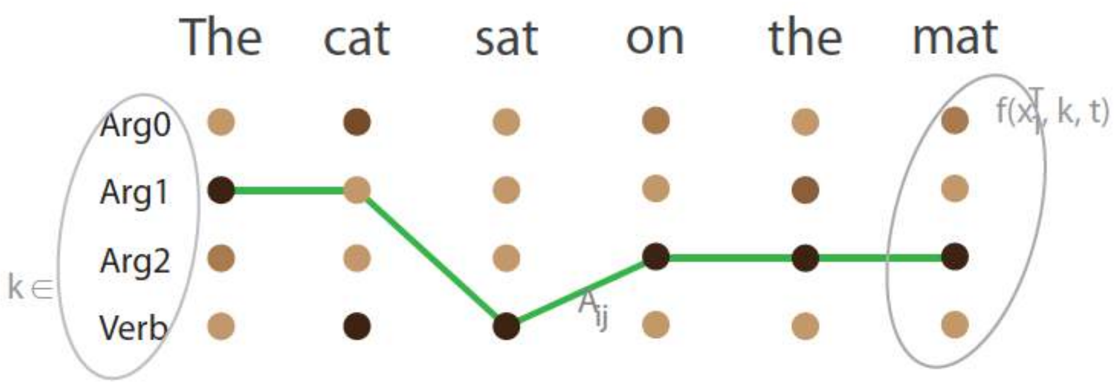
$$s([x]_1^T, [i]_1^T, \tilde{\theta}) = \sum_{t=1}^{T} \left(A_{[i]_{t-1}[i]_t} + f([x]_1^T, [i]_t, t, \theta) \right)$$

  - where $A_{[i][j]}$ is a transition score for jumping from tag *i* to *j*

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. J. Mach. Learn. Res. 12 (November 2011), 2493-2537.

Sentence-Level Log-Likelihood

- Decoding: finding the max scored path
  - Viterbi algorithm



Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. J. Mach. Learn. Res. 12 (November 2011), 2493-2537.

Results

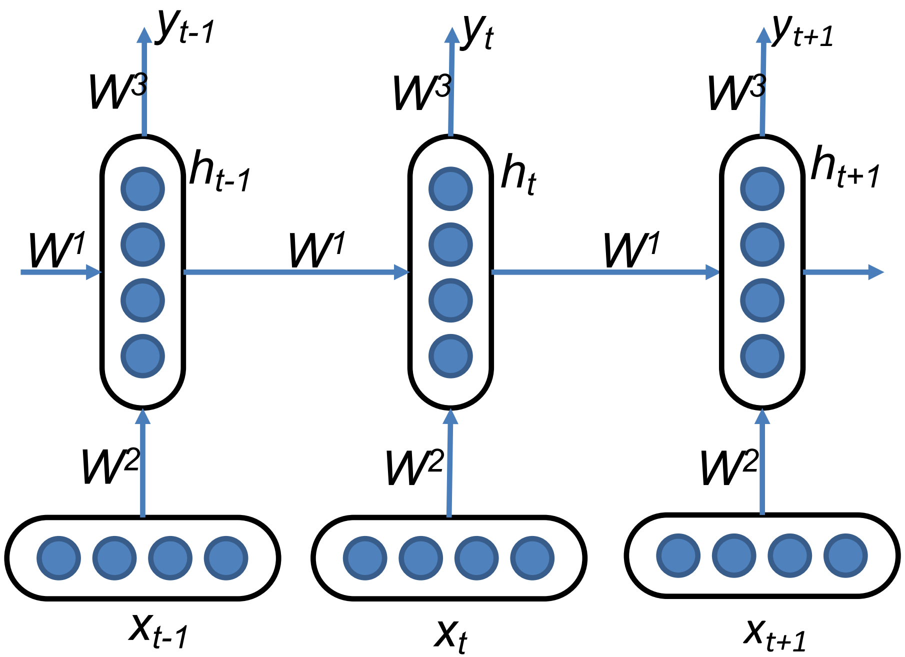
Approach	POS (PWA)	Chunking (F1)	NER (F1)	SRL (F1)
Benchmark Systems	97.24	94.29	89.31	77.92
NN+WLL	96.31	89.13	79.53	55.40
NN+SLL	96.37	90.33	81.47	70.99

- SLL helps, but fair performance for POS

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. J. Mach. Learn. Res. 12 (November 2011), 2493-2537.

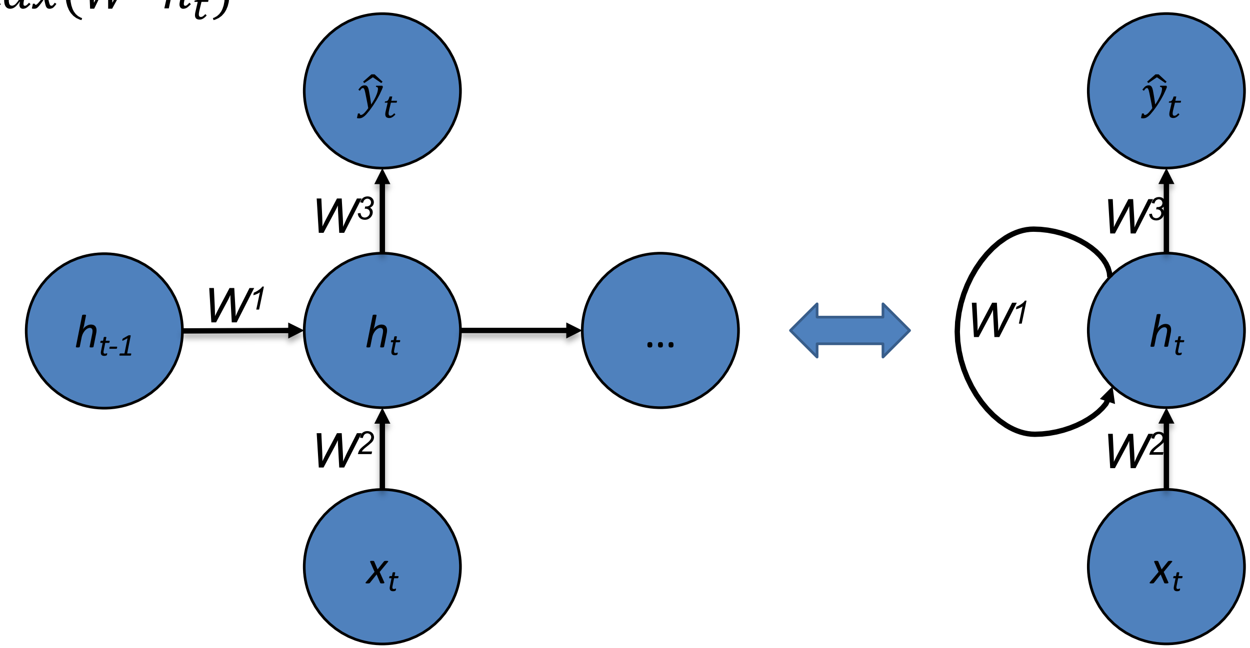
Recurrent Neural Networks (RNNs)

- Condition the neural network on all previous inputs
- RAM requirement only scales with number of inputs



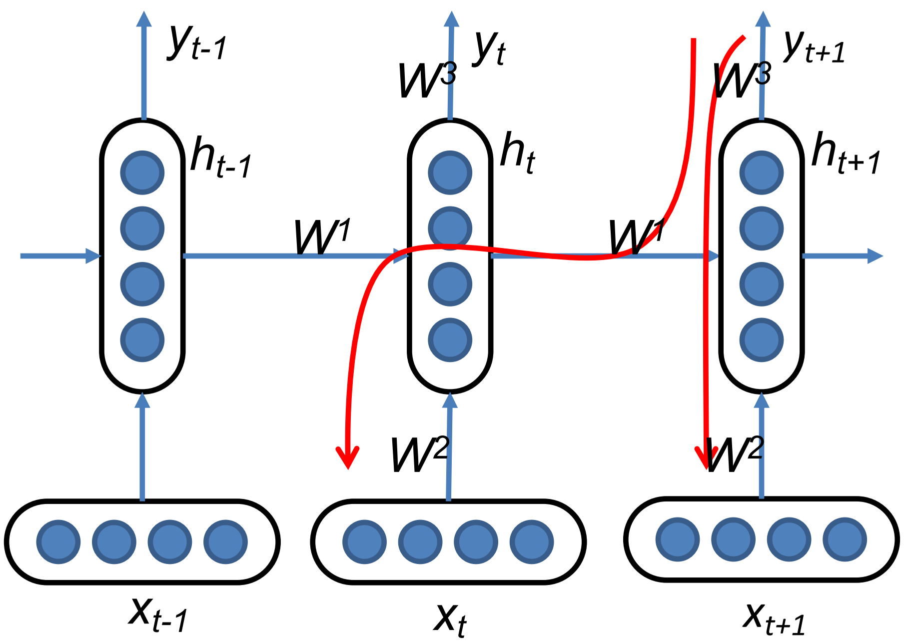
Recurrent Neural Networks (RNNs)

- At a single time step t
  - $h_t = \tanh(W^1 h_{t-1} + W^2 x_t)$
  - $\hat{y}_t = softmax(W^3 h_t)$



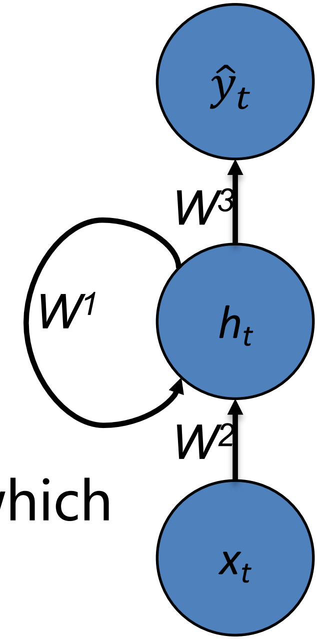
Training RNNs is hard

- Ideally inputs from many time steps ago can modify output y
- For example, with 2 time steps



BackPropagation Through Time (BPTT)

- Total error is the sum of each error at time step t
  - $\frac{\partial E}{\partial W} = \sum_{t=1}^{T} \frac{\partial E_t}{\partial W}$
- $\frac{\partial E_t}{\partial W^3} = \frac{\partial E_t}{\partial y_t} \frac{\partial y_t}{\partial W^3}$ is easy to be calculated
- But to calculate $\frac{\partial E_t}{\partial W^1} = \frac{\partial E_t}{\partial y_t} \frac{\partial y_t}{\partial h_t} \frac{\partial h_t}{\partial W^1}$ is hard (also for W^2)
- Because $h_t = \tanh(W^1 h_{t-1} + W^2 x_t)$ depends on h_{t-1}, which depends on W^1 and h_{t-2}, and so on.
- So $\frac{\partial E_t}{\partial W^1} = \sum_{k=1}^{t} \frac{\partial E_t}{\partial y_t} \frac{\partial y_t}{\partial h_t} \frac{\partial h_t}{\partial h_k} \frac{\partial h_k}{\partial W^1}$



BackPropagation Through Time (BPTT)

- Use the same as the backpropagation algorithm as we use in deep feedforward NN, but summing up the gradients for W^1
- BPTT is just a **fancy name** for standard backpropagation on an unrolled RNN
- $\frac{\partial E}{\partial W^1} = \sum_{t=1}^{T} \frac{\partial E_t}{\partial W^1}$
- $\frac{\partial E_t}{\partial W^1} = \sum_{k=1}^{t} \frac{\partial E_t}{\partial y_t} \frac{\partial y_t}{\partial h_t} \frac{\partial h_t}{\partial h_k} \frac{\partial h_k}{\partial W^1}$

The vanishing gradient problem

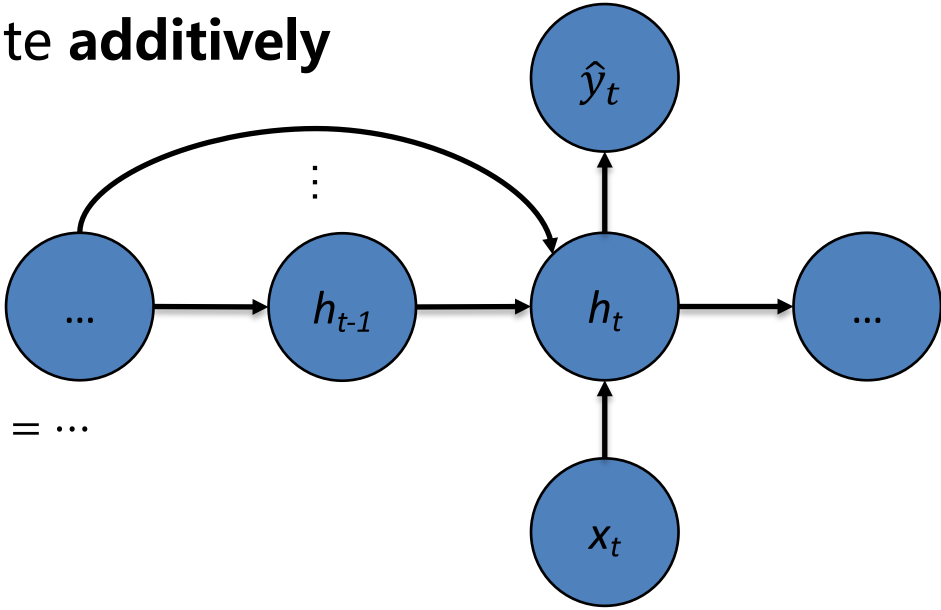
- $\frac{\partial E_t}{\partial W} = \sum_{k=1}^{t} \frac{\partial E_t}{\partial y_t} \frac{\partial y_t}{\partial h_t} \boxed{\frac{\partial h_t}{\partial h_k}} \frac{\partial h_k}{\partial W}$, $h_t = \tanh(W^1 h_{t-1} + W^2 x_t)$
- $\frac{\partial h_t}{\partial h_k} = \prod_{j=k+1}^{t} \frac{\partial h_j}{\partial h_{j-1}} = \prod_{j=k+1}^{t} W^1 \text{diag}[\tanh'(\cdots)]$
- $\left\|\frac{\partial h_t}{\partial h_{t-1}}\right\| \le \gamma \|W^1\| \le \gamma \lambda_1$
  - where γ is bound $\|\text{diag}[\tanh'(\cdots)]\|$, λ_1 is the largest singular value of W^1
- $\left\|\frac{\partial h_t}{\partial h_k}\right\| \le (\gamma \lambda_1)^{t-k}$
- This can become very small or very large quickly → **Vanishing** or **exploding** gradient
  - Trick for exploding gradient: clipping trick (set a threshold)

A "Solution"

- Intuition
  - Ensure $\gamma\lambda_1 \geq 1$ → to prevent vanishing gradients
- So …
  - Proper initialization of the W
  - To use ReLU instead of tanh or sigmoid activation functions

A better “solution”

- Recall the original transition equation
  - $h_t = \tanh(W^1 h_{t-1} + W^2 x_t)$
- We can instead update the state **additively**
  - $u_t = \tanh(W^1 h_{t-1} + W^2 x_t)$
  - $h_t = h_{t-1} + u_t$
  - then, $\left\|\frac{\partial h_t}{\partial h_{t-1}}\right\| = 1 + \left\|\frac{\partial u_t}{\partial h_{t-1}}\right\| \geq 1$
  - On the other hand
    - $h_t = h_{t-1} + u_t = h_{t-2} + u_{t-1} + u_t = \cdots$



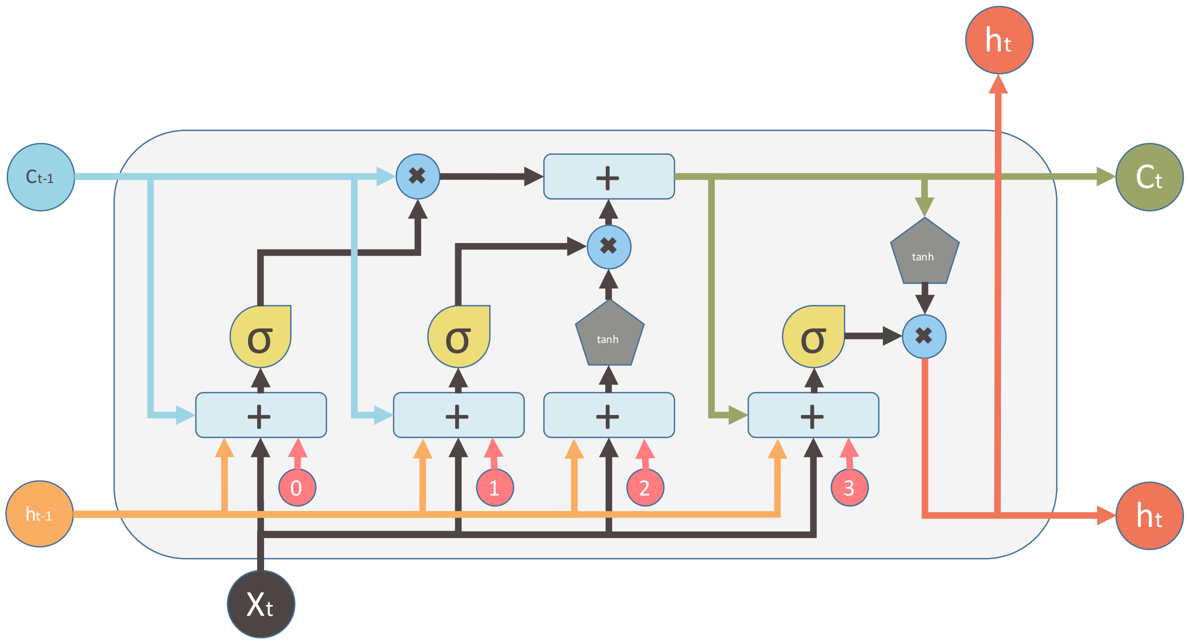
A better “solution” (cont.)

- Interpolate between old state and new state (“choosing to **forget**”)
  - $f_t = \sigma\left(W^f x_t + U^f h_{t-1}\right)$
  - $h_t = f_t \odot h_{t-1} + (1 - f_t) \odot u_t$
- Introduce a separate **input gate** i_t
  - $i_t = \sigma\left(W^i x_t + U^i h_{t-1}\right)$
  - $h_t = f_t \odot h_{t-1} + i_t \odot u_t$
- Selectively expose memory cell c_t with an **output gate** o_t
  - $o_t = \sigma(W^o x_t + U^o h_{t-1})$
  - $c_t = f_t \odot c_{t-1} + i_t \odot u_t$
  - $h_t = o_t \odot \tanh(c_t)$

Long Short-Term Memory (LSTM)

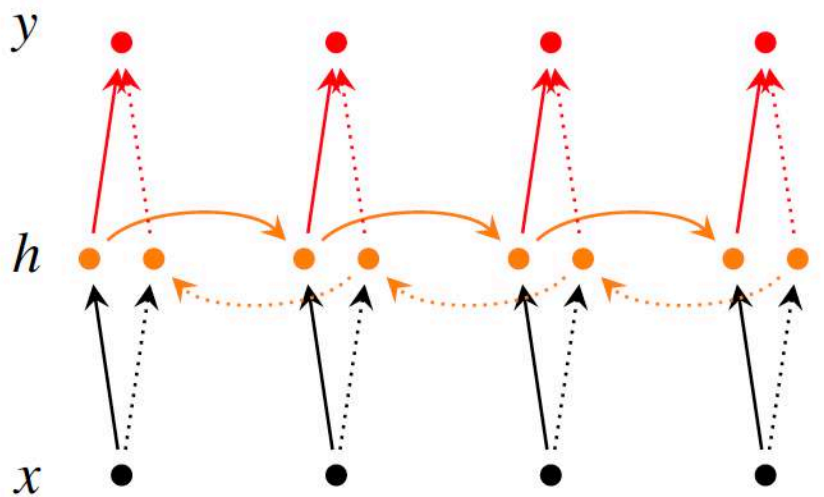
- Hochreiter & Schmidhuber, 1997
- LSTM = additive updates + gating

$$u_t = \tanh\left(W h_{t-1} + V x_t\right)$$
$$f_t = \text{sigmoid}\left(W_f h_{t-1} + V_f x_t\right)$$
$$i_t = \text{sigmoid}\left(W_i h_{t-1} + V_i x_t\right)$$
$$o_t = \text{sigmoid}\left(W_o h_{t-1} + V_o x_t\right)$$
$$c_t = f_t \odot c_{t-1} + i_t \odot u_t$$
$$h_t = o_t \odot \tanh(c_t)$$
$$y_t = U h_t$$

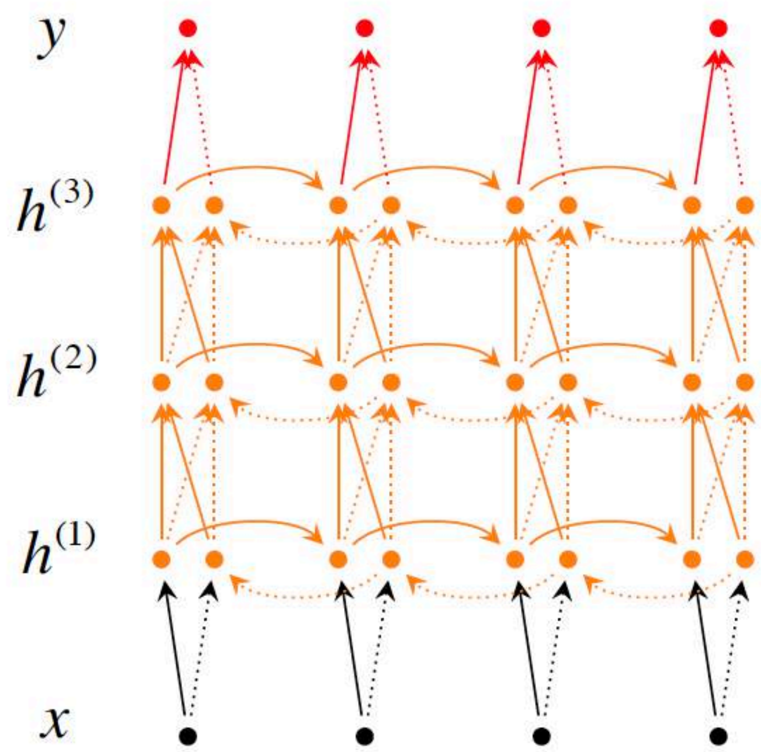


More RNNs

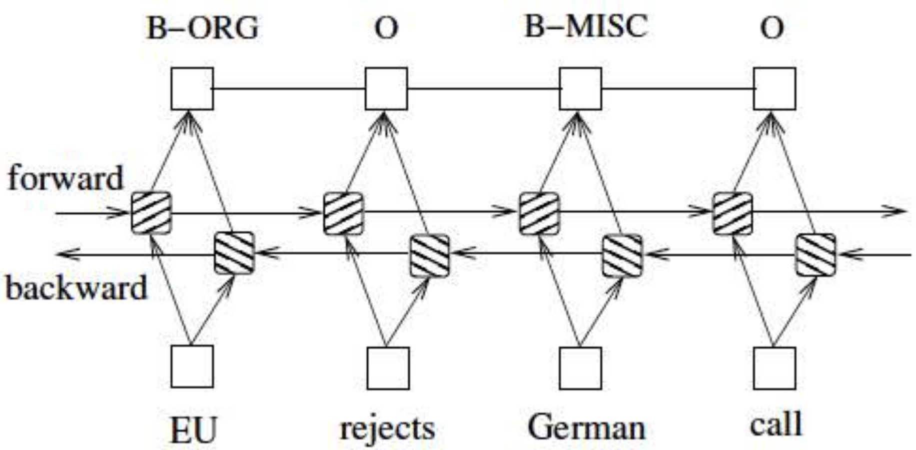
- Bidirectional RNN



- Deep Bidirectional



Bi-LSTM-CRF



Algorithm 1 Bidirectional LSTM CRF model training procedure

```
for each epoch do
    for each batch do
        1) bidirectional LSTM-CRF model forward pass:
            forward pass for forward state LSTM
            forward pass for backward state LSTM
        2) CRF layer forward and backward pass
        3) bidirectional LSTM-CRF model backward pass:
            backward pass for forward state LSTM
            backward pass for backward state LSTM
        4) update parameters
    end for
end for
```

Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional LSTM-CRF models for sequence tagging. CoRR, abs/1508.01991, 2015.

Results

		POS	Chunking CoNLL2000	NER CoNLL2003
Random	Conv-CRF (Collobert et al., 2011)	96.37	90.33	81.47
	LSTM	97.10	92.88	79.82
	BI-LSTM	97.30	93.64	81.11
	CRF	97.30	93.69	83.02
	LSTM-CRF	**97.45**	93.80	84.10
	BI-LSTM-CRF	97.43	**94.13**	**84.26**
Senna	Conv-CRF (Collobert et al., 2011)	97.29	94.32	88.67 (89.59)
	LSTM	97.29	92.99	83.74
	BI-LSTM	97.40	93.92	85.17
	CRF	97.45	93.83	86.13
	LSTM-CRF	97.54	94.27	88.36
	BI-LSTM-CRF	**97.55**	**94.46**	**88.83 (90.10)**

Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional LSTM-CRF models for sequence tagging. CoRR, abs/1508.01991, 2015.

BI-LSTM-CRF for SRL

- End-to-end tagging r
  - 8 layer bi-directional L
  - No parsing features
- Features
  - Argument
  - Predicate
  - Predicate-context
  - Region-mark

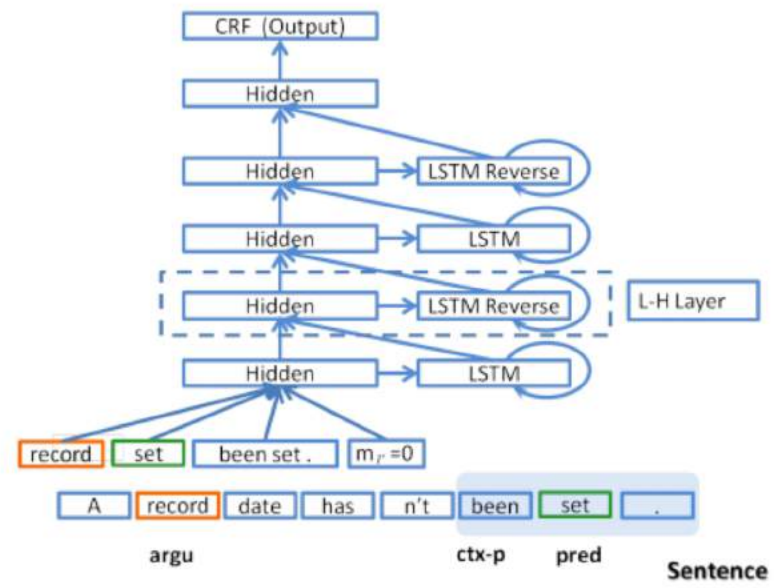
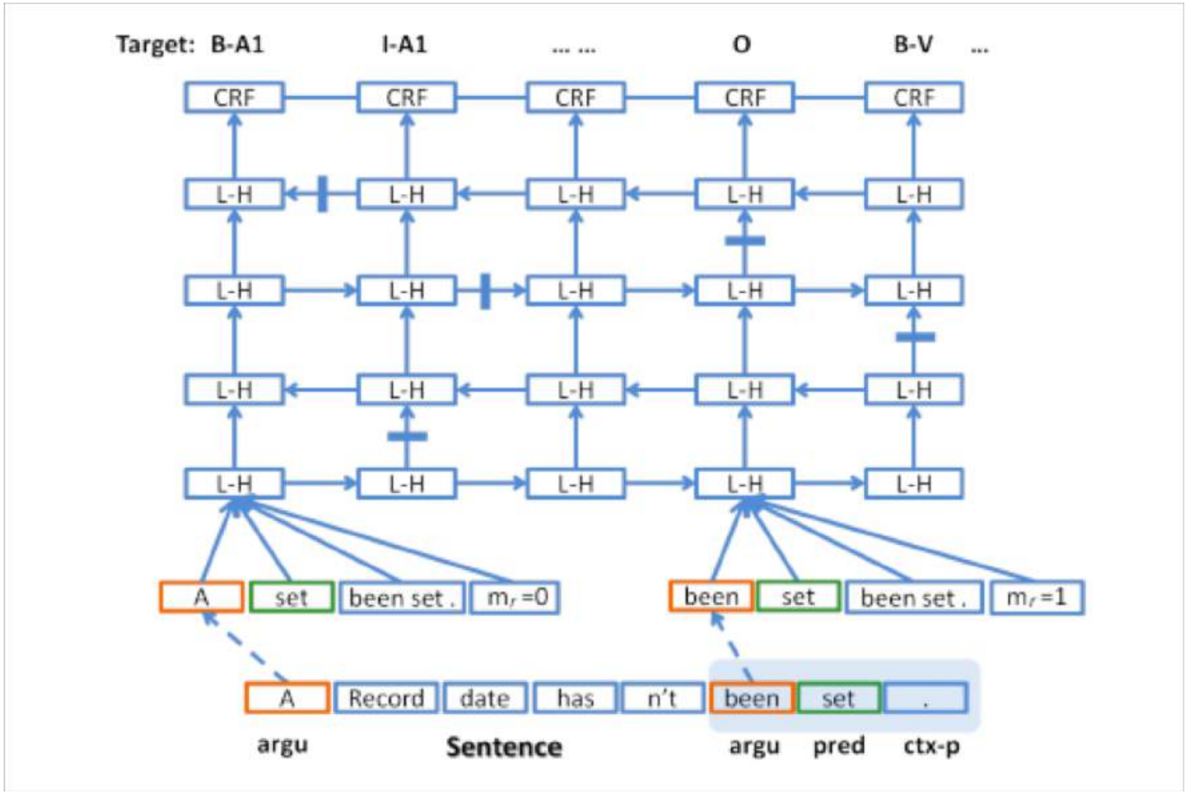


Figure 2: DB-LSTM network.Shadow part denote the predicate context within length 1.

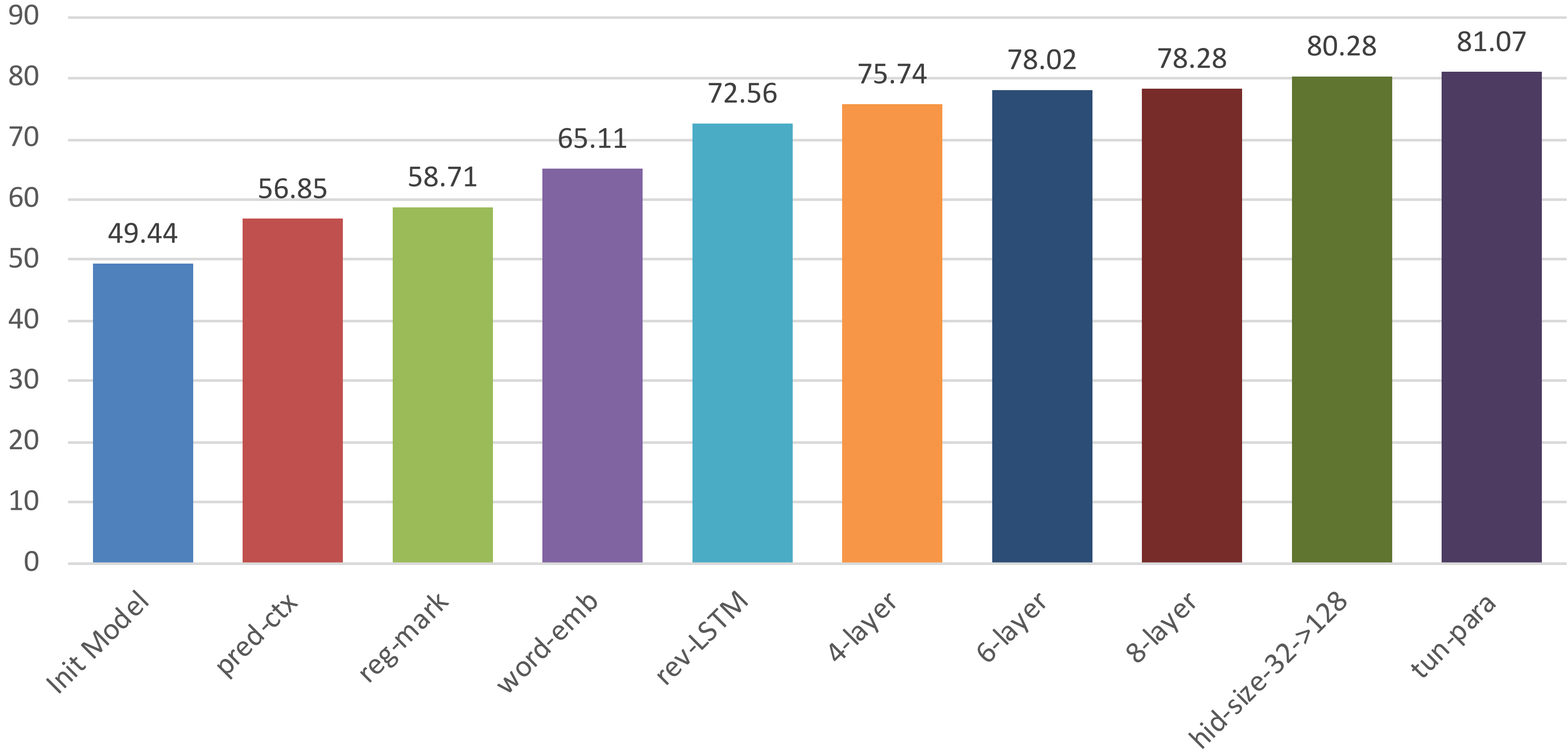
Jie Zhou and Wei Xu. (2015). End-to-end learning of semantic role labeling using recurrent neural networks. ACL.

Temporal Expanded



Jie Zhou and Wei Xu. (2015). End-to-end learning of semantic role labeling using recurrent neural networks. ACL.

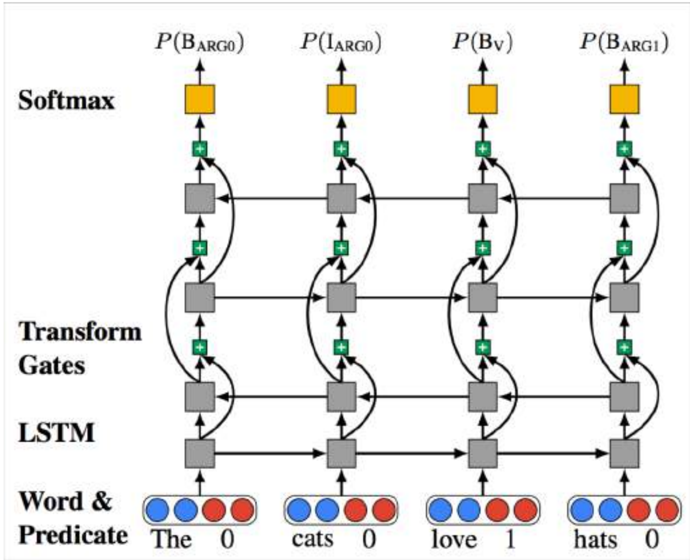
Results



Jie Zhou and Wei Xu. (2015). End-to-end learning of semantic role labeling using recurrent neural networks. ACL.

Deep SRL

- A deep **highway** BiLSTM architecture with constraints
  - 8 BiLSTM layers (4 forward LSTMs and 4 reversed LSTMs)



Luheng He, Kenton Lee, Mike Lewis and Luke Zettlemoyer. Deep Semantic Role Labeling: What Works and What's Next. ACL 2017.

Results

- New state-of-the-art results

Method	Development				WSJ Test				Brown Test				Combined
	P	R	F1	Comp.	P	R	F1	Comp.	P	R	F1	Comp.	F1
Ours (PoE)	**83.1**	**82.4**	**82.7**	**64.1**	**85.0**	**84.3**	**84.6**	**66.5**	**74.9**	**72.4**	**73.6**	**46.5**	**83.2**
Ours	81.6	81.6	81.6	62.3	83.1	83.0	83.1	64.3	72.9	71.4	72.1	44.8	81.6
Zhou	79.7	79.4	79.6	-	82.9	82.8	82.8	-	70.7	68.2	69.4	-	81.1
FitzGerald (Struct.,PoE)	81.2	76.7	78.9	55.1	82.5	78.2	80.3	57.3	74.5	70.0	72.2	41.3	-
Täckström (Struct.)	81.2	76.2	78.6	54.4	82.3	77.6	79.9	56.0	74.3	68.6	71.3	39.8	-
Toutanova (Ensemble)	-	-	78.6	58.7	81.9	78.8	80.3	60.1	-	-	68.8	40.8	-
Punyakanok (Ensemble)	80.1	74.8	77.4	50.7	82.3	76.8	79.4	53.8	73.4	62.9	67.8	32.3	77.9

Luheng He, Kenton Lee, Mike Lewis and Luke Zettlemoyer. Deep Semantic Role Labeling: What Works and What's Next. ACL 2017.

Part 2.2: Neural Semi-CRF

Segmentation Models

- Tagging models cannot extract segment information
  - E.g. the length of a segment
- Some tagging problems can be naturally modeled into segmentation task
  - E.g. word segmentation, named entity recognition

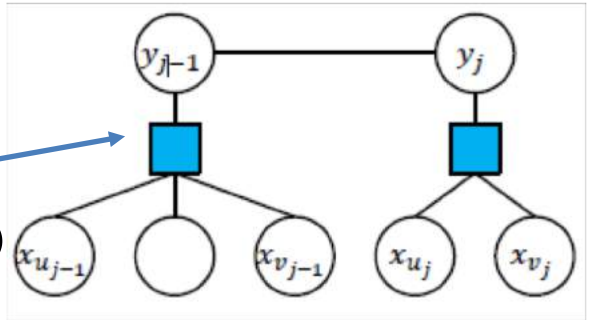
浦东开发与建设 → 浦东 / 开发 / 与 / 建设
Pudong development and construction

Yijia Liu, Wanxiang Che, Jiang Guo, Bing Qin, Ting Liu. (2016). Exploring Segment Representations for Neural Segmentation Models. IJCAI.

Semi-CRF

- A solution
  - Semi-Markov CRF [Sarawagi and Cohen, 2004]
  - Modeling segments directly
  - $p(\mathbf{s}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\{W \cdot G(\mathbf{x}, \mathbf{s})\}$

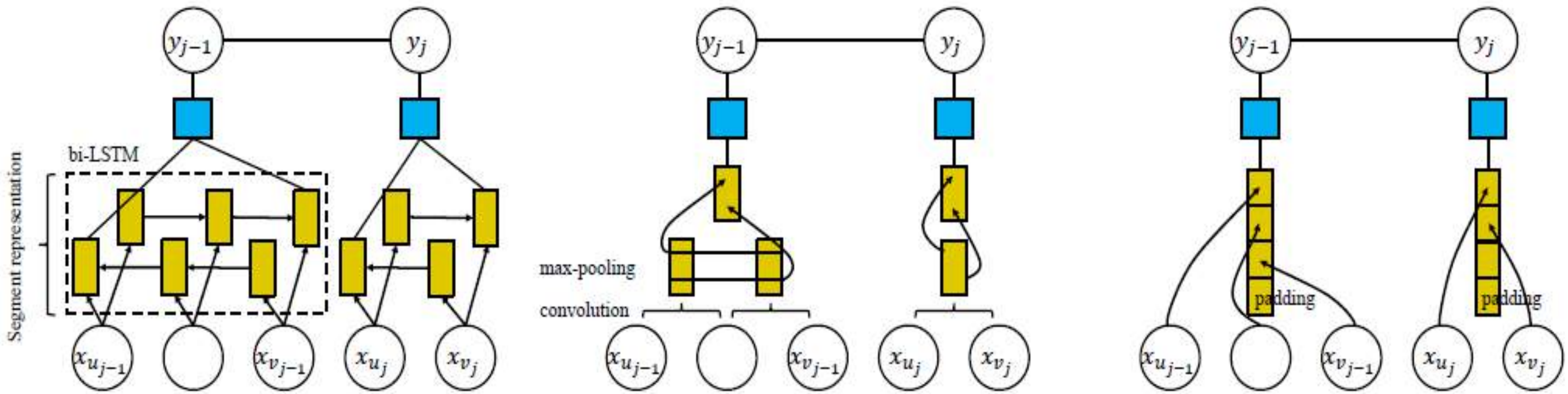
Feature extraction G(x,s)



Can we represent segments with vectors?

Yijia Liu, Wanxiang Che, Jiang Guo, Bing Qin, Ting Liu. (2016). Exploring Segment Representations for Neural Segmentation Models. IJCAI.

Compositional Segment Representation



(b) SRNN (c) SCNN (d) SCONCATE

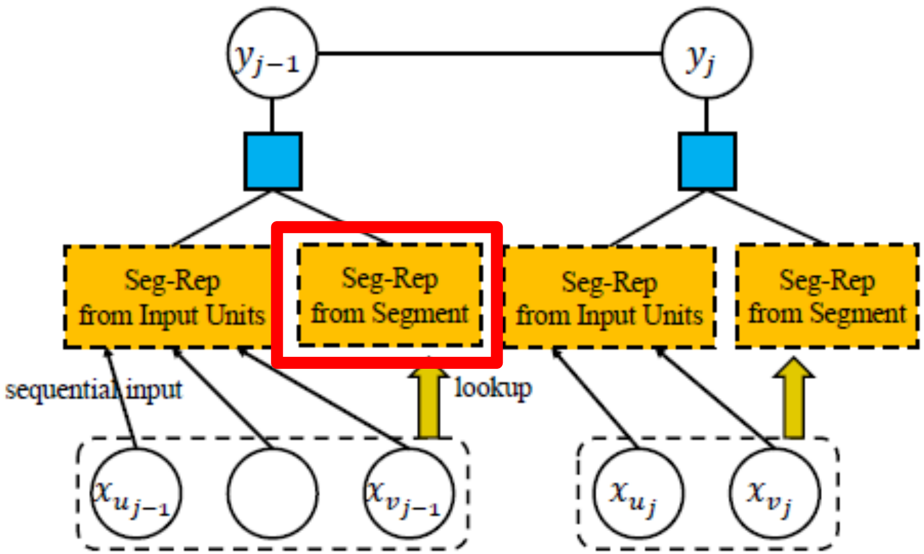
Yijia Liu, Wanxiang Che, Jiang Guo, Bing Qin, Ting Liu. (2016). Exploring Segment Representations for Neural Segmentation Models. IJCAI.

Results

	model	NER CoNLL03 dev	NER CoNLL03 test	CWS CTB6 dev	CWS CTB6 test	CWS PKU dev	CWS PKU test	CWS MSR dev	CWS MSR test	spd
baseline	NN-LABELER	93.03	88.62	93.70	93.06	93.57	92.99	93.22	93.79	**3.30**
	NN-CRF	**93.06**	**89.08**	94.33	93.65	94.09	93.28	93.81	94.17	2.72
	SPARSE-CRF	88.87	83.43	**95.68**	**95.08**	**95.85**	**95.06**	**96.09**	**96.54**	
neural semi-CRF	SRNN	92.97	88.63	94.56	94.06	94.86	93.91	94.38	95.21	0.62
	SCONCATE	92.96	89.07	94.34	93.96	94.41	93.57	94.05	94.53	1.08
	SCNN	91.53	87.68	87.82	87.51	79.64	80.75	85.04	85.79	1.46

Yijia Liu, Wanxiang Che, Jiang Guo, Bing Qin, Ting Liu. (2016). Exploring Segment Representations for Neural Segmentation Models. IJCAI.

Segment-level Representation



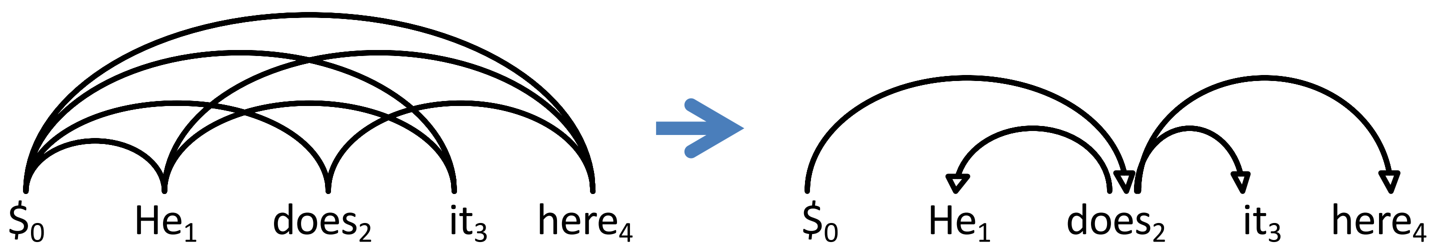
model	CoNLL03	CTB6	PKU	MSR
NN-LABELER	88.62	93.06	92.99	93.79
NN-CRF	89.08	93.65	93.28	94.17
SPARSE-CRF	83.43	95.08	95.06	96.54
SRNN	88.63	94.06	93.91	95.21
+SEMB-HETERO	89.59	**95.48**	95.60	97.39
	+0.96	*+1.42*	*+1.69*	*+2.18*
SCONCATE	89.07	93.96	93.57	94.53
+SEMB-HETERO	**89.77**	95.42	**95.67**	**97.58**
	+0.70	*+1.43*	*+2.10*	*+3.05*

Yijia Liu, Wanxiang Che, Jiang Guo, Bing Qin, Ting Liu. (2016). Exploring Segment Representations for Neural Segmentation Models. IJCAI.

Part 2.3: Graph-based Dependency Parsing

Graph-based Dependency Parsing

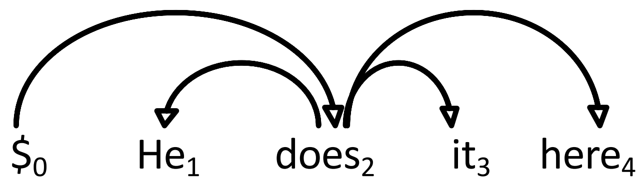
- Find the highest scoring tree from a complete dependency graph



$$Y^* = \arg\max_{Y \in \Phi(X)} score(X, Y)$$

First-order as an Example

- The first-order graph-based method assumes that arcs in a tree are independent from each other (arc-factorization)

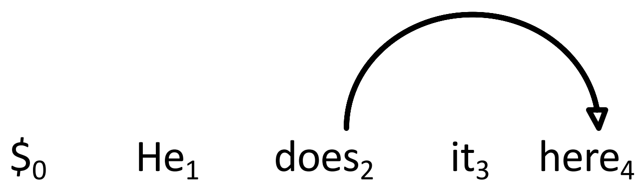


$$score(X,Y) = \sum_{(h,m)\in Y} score(X,h,m)$$

How to Score an Arc

- Given an sentence, how to determine the score of each arc?

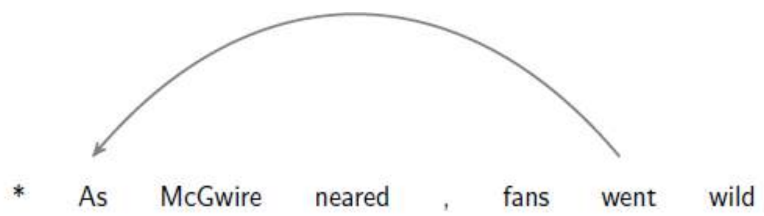
$$score(2,4) = ?$$



- Feature based representation: an arc is represented as a feature vector **f**(2,4)

$$score(2,4) = \mathbf{w} \cdot \mathbf{f}(2,4)$$

Features for an Arc

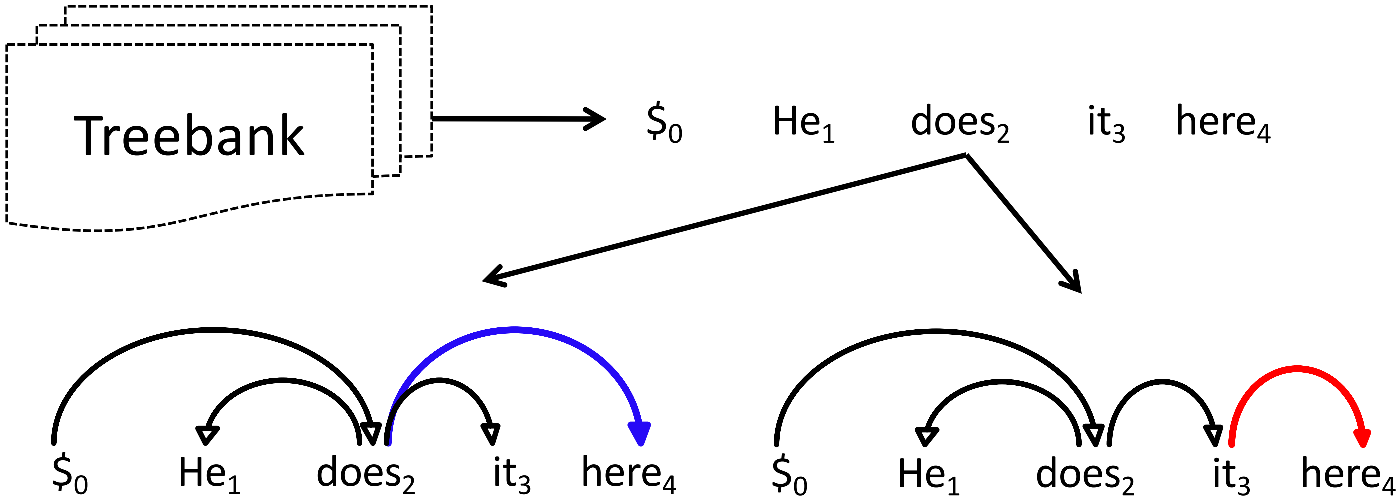


[went]	[VBD]	[As]	[ADP]	[went]
[VERB]	[As]	[IN]	[went, VBD]	[As, ADP]
[went, As]	[VBD, ADP]	[went, VERB]	[As, IN]	[went, As]
[VERB, IN]	[VBD, As, ADP]	[went, As, ADP]	[went, VBD, ADP]	[went, VBD, As]
[ADJ, *, ADP]	[VBD, *, ADP]	[VBD, ADJ, ADP]	[VBD, ADJ, *]	[NNS, *, ADP]
[NNS, VBD, ADP]	[NNS, VBD, *]	[ADJ, ADP, NNP]	[VBD, ADP, NNP]	[VBD, ADJ, NNP]
[NNS, ADP, NNP]	[NNS, VBD, NNP]	[went, left, 5]	[VBD, left, 5]	[As, left, 5]
[ADP, left, 5]	[VERB, As, IN]	[went, As, IN]	[went, VERB, IN]	[went, VERB, As]
[JJ, *, IN]	[VERB, *, IN]	[VERB, JJ, IN]	[VERB, JJ, *]	[NOUN, *, IN]
[NOUN, VERB, IN]	[NOUN, VERB, *]	[JJ, IN, NOUN]	[VERB, IN, NOUN]	[VERB, JJ, NOUN]
[NOUN, IN, NOUN]	[NOUN, VERB, NOUN]	[went, left, 5]	[VERB, left, 5]	[As, left, 5]
[IN, left, 5]	[went, VBD, As, ADP]	[VBD, ADJ, *, ADP]	[NNS, VBD, *, ADP]	[VBD, ADJ, ADP, NNP]
[NNS, VBD, ADP, NNP]	[went, VBD, left, 5]	[As, ADP, left, 5]	[went, As, left, 5]	[VBD, ADP, left, 5]
[went, VERB, As, IN]	[VERB, JJ, *, IN]	[NOUN, VERB, *, IN]	[VERB, JJ, IN, NOUN]	[NOUN, VERB, IN, NOUN]
[went, VERB, left, 5]	[As, IN, left, 5]	[went, As, left, 5]	[VERB, IN, left, 5]	[VBD, As, ADP, left, 5]
[went, As, ADP, left, 5]	[went, VBD, ADP, left, 5]	[went, VBD, As, left, 5]	[ADJ, *, ADP, left, 5]	[VBD, *, ADP, left, 5]
[VBD, ADJ, ADP, left, 5]	[VBD, ADJ, *, left, 5]	[NNS, *, ADP, left, 5]	[NNS, VBD, ADP, left, 5]	[NNS, VBD, *, left, 5]
[ADJ, ADP, NNP, left, 5]	[VBD, ADP, NNP, left, 5]	[VBD, ADJ, NNP, left, 5]	[NNS, ADP, NNP, left, 5]	[NNS, VBD, NNP, left, 5]
[VERB, As, IN, left, 5]	[went, As, IN, left, 5]	[went, VERB, IN, left, 5]	[went, VERB, As, left, 5]	[JJ, *, IN, left, 5]
[VERB, *, IN, left, 5]	[VERB, JJ, IN, left, 5]	[VERB, JJ, *, left, 5]	[NOUN, *, IN, left, 5]	[NOUN, VERB, IN, left, 5]

Decoding for first-order model

- Maximum Spanning Tree (MST) Algorithm
- Eisner (2000) described a **dynamic programming** based decoding algorithm for bilexical grammar
- McDonald+ (2005) applied this algorithm to the search problem of the first-order model

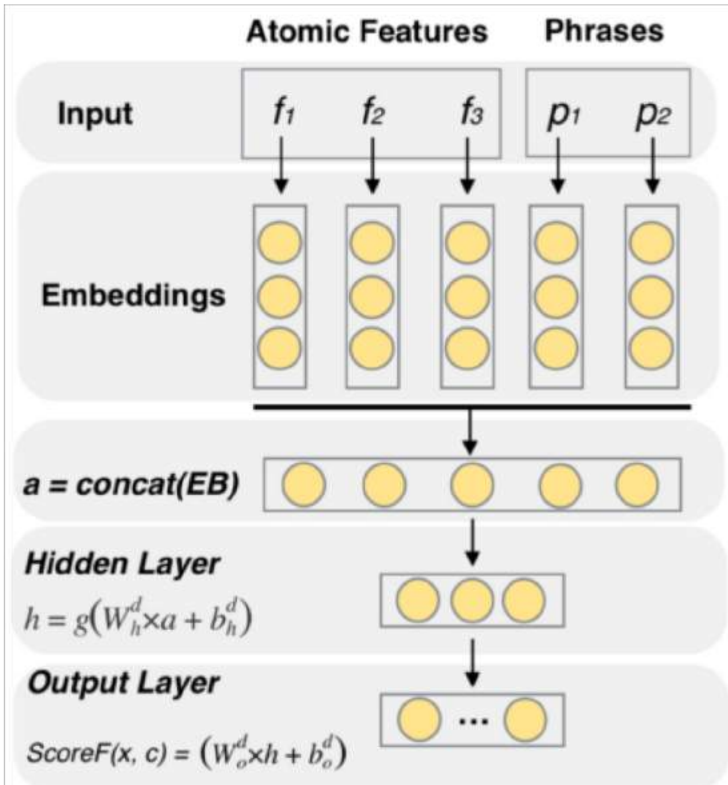
Online learning w



Gold-standard parse Y^+ 1-best parse Y^- with $\mathbf{w}^{(k)}$

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \mathbf{f}(X,Y^+) - \mathbf{f}(X,Y^-)$$

NN for Graph-based Parsing



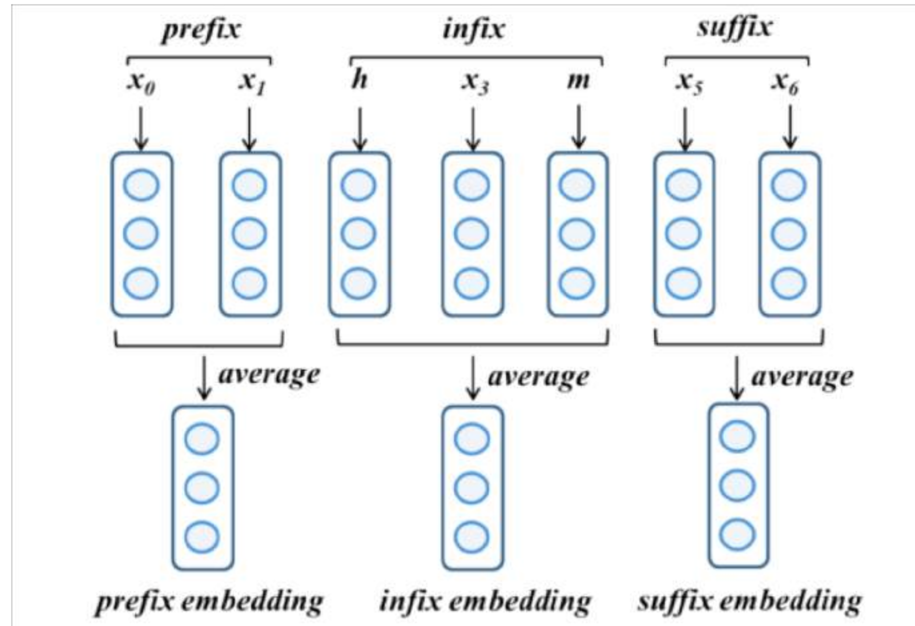
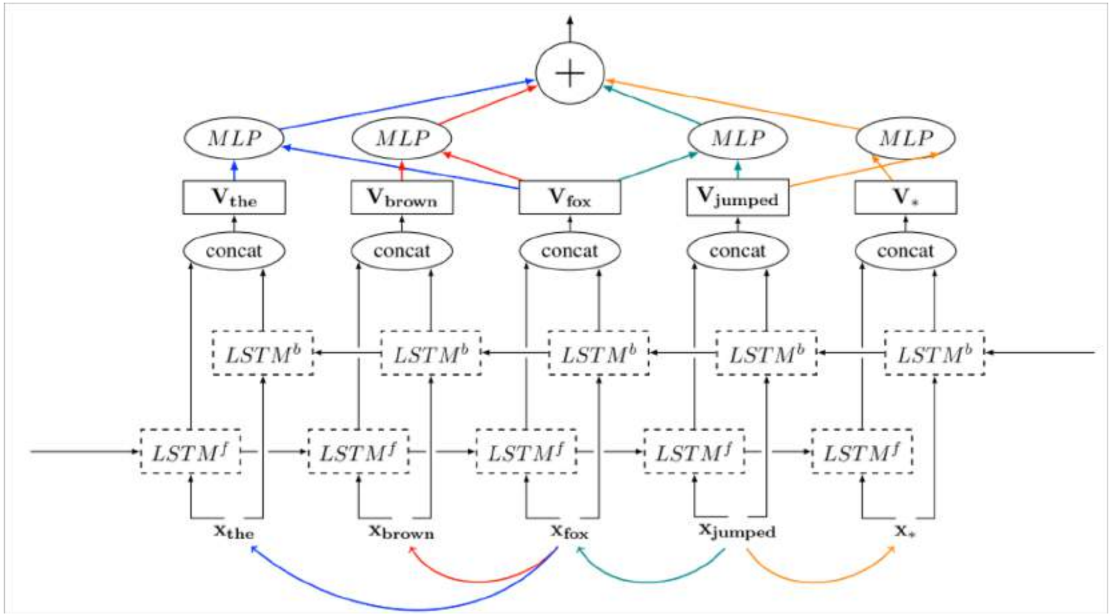


Figure 3: Illustration for phrase embeddings. h, m and x_0 to x_6 are words in the sentence.

Pei, W., Ge, T., & Chang, B. (2015). An Effective Neural Network Model for Graph-based Dependency Parsing. ACL.

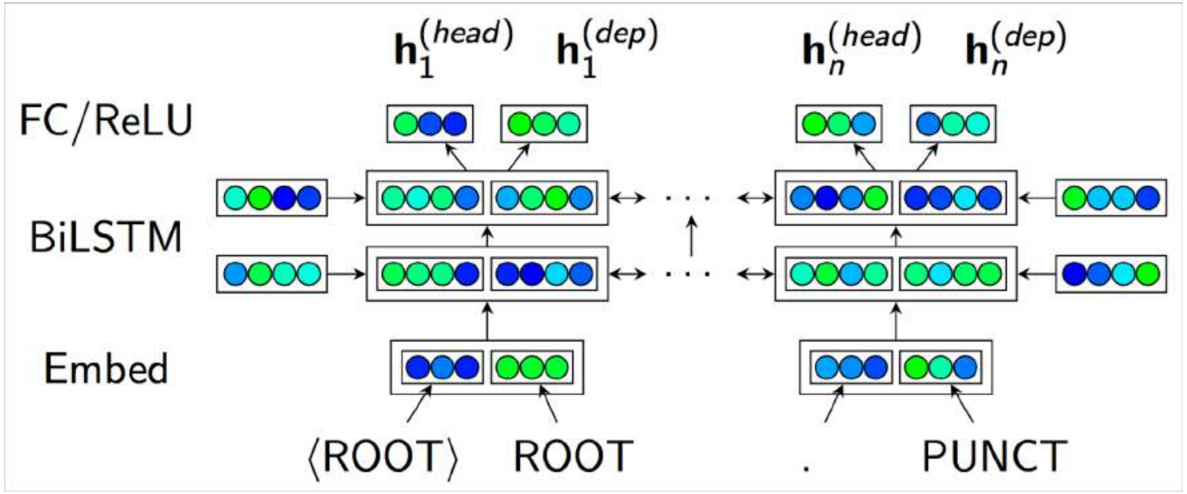
BI-LSTM for Graph-based Parsing-I

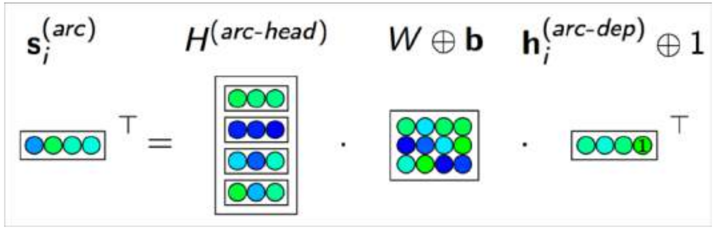
- Each dependency arc in a sentence is scored using MLP that is fed the BI-LSMT encoding of the words at the arc’ s end points



Kiperwasser, E., & Goldberg, Y. (2016). Simple and Accurate Dependency Parsing Using Bidirectional LSTM Feature Representations. TACL.

Deep Biaffine Attention for Dependency Parsing





- Just optimize the likelihood of the head, no structured learning
- This is a local model, with global decoding using MST at the end

Timothy Dozat and Christopher D. Manning. Deep Biaffine Attention for Neural Dependency Parsing. ICLR 2017.

Results

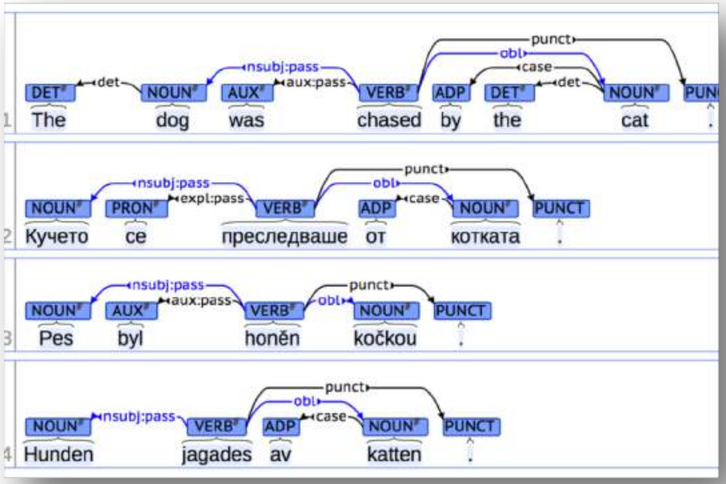
Type	Model	English PTB-SD 3.3.0 UAS	English PTB-SD 3.3.0 LAS	Chinese PTB 5.1 UAS	Chinese PTB 5.1 LAS
Transition	Ballesteros et al. (2016)	93.56	91.42	87.65	86.21
	Andor et al. (2016)	94.61	92.79	–	–
	Kuncoro et al. (2016)	**95.8**	**94.6**	–	–
Graph	Kiperwasser & Goldberg (2016)	93.9	91.9	87.6	86.1
	Cheng et al. (2016)	94.10	91.49	88.1	85.7
	Hashimoto et al. (2016)	94.67	92.90	–	–
	Deep Biaffine	95.74	94.08	**89.30**	**88.23**

- Tuning Adam

Model	Adam UAS	LAS
$\beta_2 = .9$	**95.75**	**94.22**
$\beta_2 = .999$	95.53*	93.91*

CoNLL 2018 Shared Task

- Multilingual Parsing from Raw Text to Universal Dependencies
  - http://universaldependencies.org/conll18/
- 82 test sets from 57 languages
  - 61 of the 82 treebanks are large
  - The other 21 treebanks are small
    - 7 treebanks have training data of a still reasonable size
    - 5 are extra test sets in languages where another large treebank exists
    - 9 are low-resource languages with no training data available



Our CoNLL 2018 Shared Task System

- Rank 1st according to LAS
- Baseline model:
  - Dozat et al. (2017)
- Winning strategies for parser:
  - ELMo: +0.8
  - Ensemble: +0.6
  - Treebank Concat.: +0.4
    (estimated on Dev set.)

LAS Ranking

1.	HIT-SCIR (Harbin)	75.84 ± 0.14	[OK]	(p<0.001)
2.	TurkuNLP (Turku)	73.28 ± 0.14	[OK]	(p=0.039)
3-5.	UDPipe Future (Praha)	73.11 ± 0.13	[OK]	(p=0.221)
3-5.	LATTICE (Paris)	73.02 ± 0.14	[OK]	(p=0.461)
3-5.	ICS PAS (Warszawa)	73.02 ± 0.14	[OK]	(p<0.001)
6.	CEA LIST (Paris)	72.56 ± 0.14	[OK]	(p=0.036)
7-8.	Uppsala (Uppsala)	72.37 ± 0.15	[OK]	(p=0.191)
7-8.	Stanford (Stanford)	72.29 ± 0.14	[OK]	(p<0.001)
9-10.	AntNLP (Shanghai)	70.90 ± 0.15	[OK]	(p=0.242)
9-10.	NLP-Cube (București)	70.82 ± 0.14	[OK]	(p=0.032)
11.	ParisNLP (Paris)	70.64 ± 0.14	[OK]	(p<0.001)
12.	SLT-Interactions (Bengaluru)	69.98 ± 0.14	[OK]	(p<0.001)
13.	IBM NY (Yorktown Heights)	69.11 ± 0.16	[OK]	(p<0.001)
14.	UniMelb (Melbourne)	68.66 ± 0.15	[OK]	(p=0.002)
15.	LeisureX (Shanghai)	68.31 ± 0.16	[OK]	(p<0.001)
16.	KParse (İstanbul)	66.58 ± 0.16	[OK]	(p=0.015)
17.	Fudan (Shanghai)	66.34 ± 0.15	[OK]	(p<0.001)
18.	BASELINE UDPipe 1.2 (Praha)	65.80 ± 0.15	[OK]	(p=0.048)
19.	Phoenix (Shanghai)	65.61 ± 0.16	[OK]	(p<0.001)
20.	CUNI x-ling (Praha)	64.87 ± 0.16	[OK]	(p<0.001)
21.	BOUN (İstanbul)	63.54 ± 0.15	[OK]	(p<0.001)
22.	ONLP lab (Ra'anana)	58.35 ± 0.15	[81]	(p<0.001)
23.	iParse (Pittsburgh)	55.83 ± 0.11	[65]	(p<0.001)
24.	HUJI (Yerushalayim)	53.69 ± 0.15	[80]	(p<0.001)
25.	ArmParser (Yerevan)	47.02 ± 0.11	[66]	(p<0.001)

Wanxiang Che, Yijia Liu, Yuxuan Wang, Bo Zheng, Ting Liu. Towards Better UD Parsing: Deep Contextualized Word Embeddings, Ensemble, and Treebank Concatenation. CoNLL 2018.

Deep Contextualized Word Embeddings (ELMo)

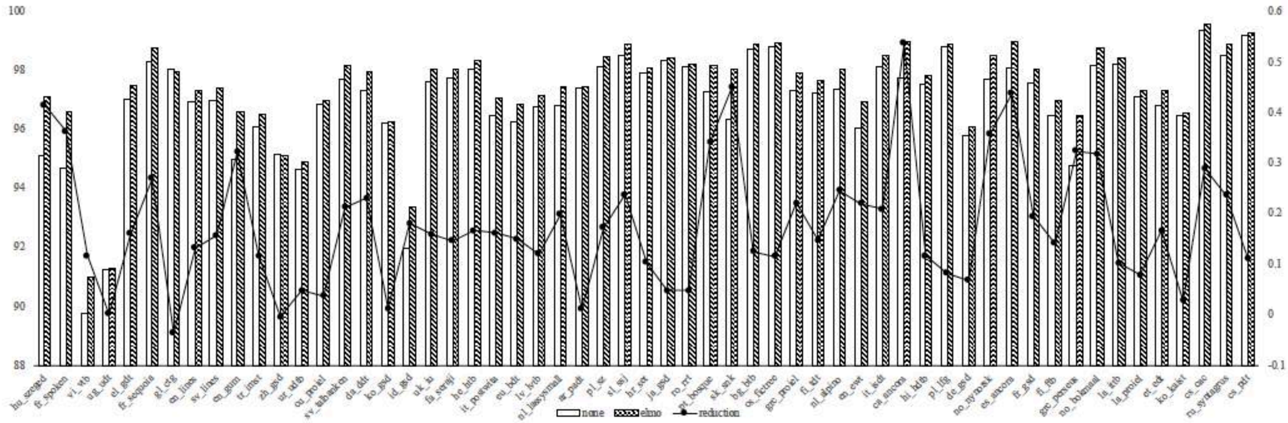
- Models pre-trained on the ImageNet are widely used for Computer Vision tasks
- What’ s the proper way to conduct pre-training for NLP?
- Leveraging Language Modeling to get pre-trained contextualized representation models
  - rely on large corpora, instead of human annotations
  - works very well -- improve the performance of existing SOTA methods a lot

Peters+. Deep contextualized word representations. NAACL 2018.

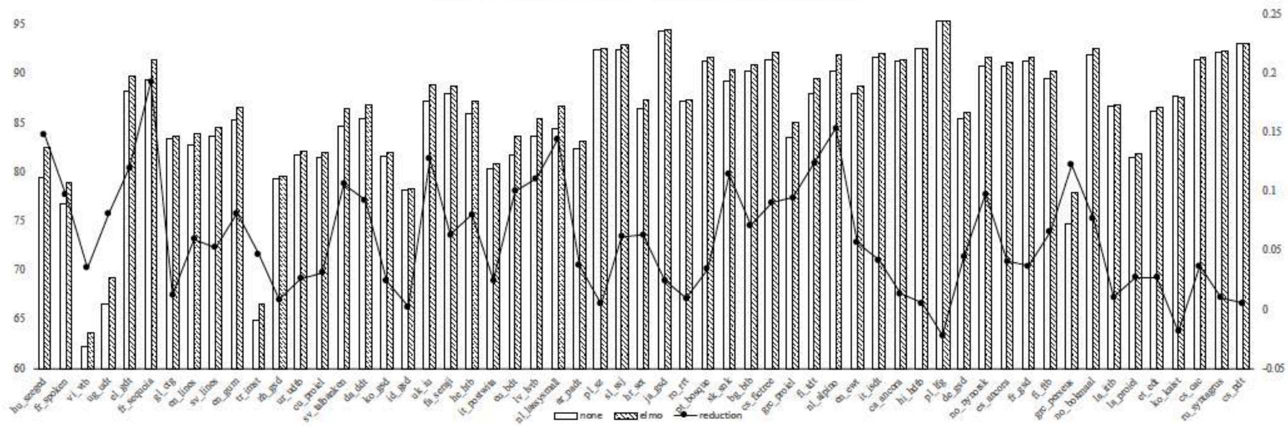
Two Extensions on ELMo

- Supporting Unicode range
- Training with *sample softmax*
  - Use a window of 8,192 surrounding words as negative samples
  - More stable training and better performance
- ELMo Training
  - Data: 20M words randomly sampled from raw text for each language
  - Time: 3 days per language on a Nvidia P100
  - We release the pre-trained ELMo
    - https://github.com/HIT-SCIR/ELMoForManyLangs

Deep Contextualized Word Embeddings (ELMo)



(a) The effects of ELMo on POS tagging



(b) The effects of ELMo on dependency parsing

Ensemble

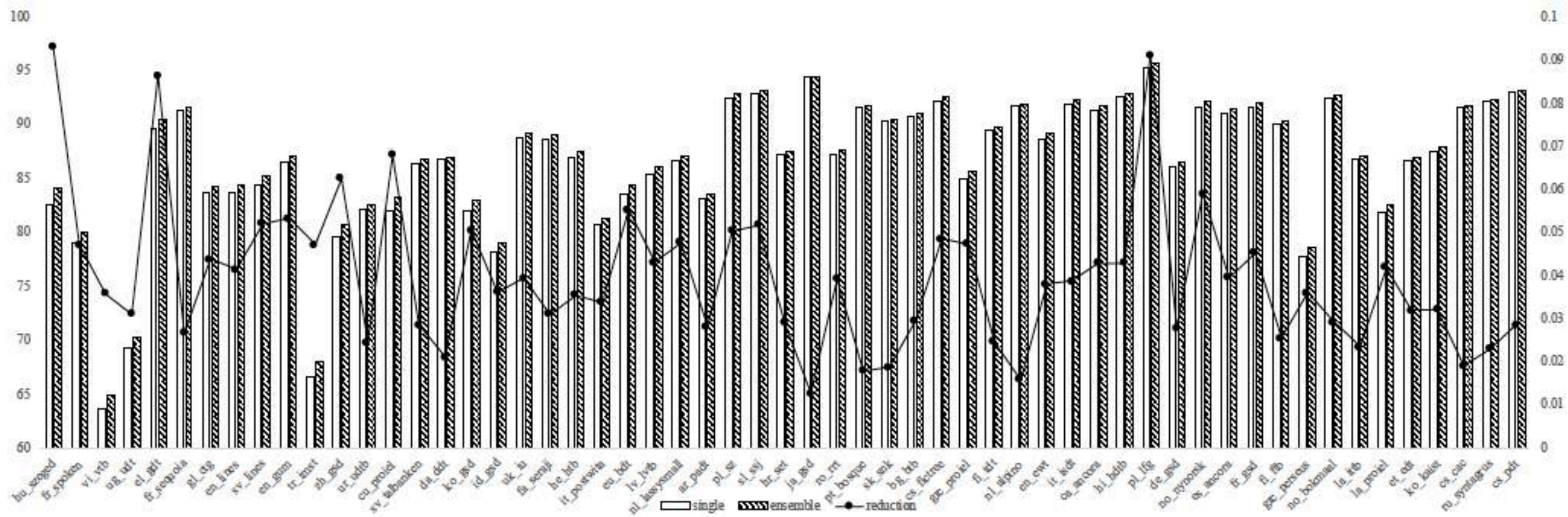


Figure 2: The effects of ensemble on dependency parsing. Treebanks are sorted according to the number of training sentences from left to right.

Part 2: Summary

- Neural Graph-based Structured Prediction
  - Sequence Labeling: Neural CRF → RNN (LSTM) → RNN+CRF
  - Segmentation: Neural Semi-CRF
  - Dependency Parsing: Neural features → LSTM → Biaffine
- Neural nets can provide continuous features in discrete structured models
- Inference and learning are almost unchanged from the purely discrete model

Part 3: Transition-based Methods

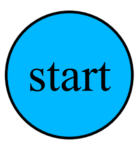
Part 3.1: Transition Systems

A transition system

- Automata
  - State
    - Start state —— an empty structure
    - End state —— the output structure
    - Intermediate states —— partially constructed structures
  - Actions
    - Change one state to another

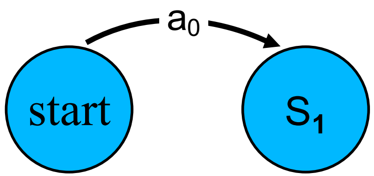
A transition system

- Automata



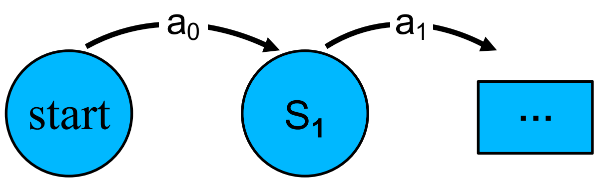
A transition system

- Automata



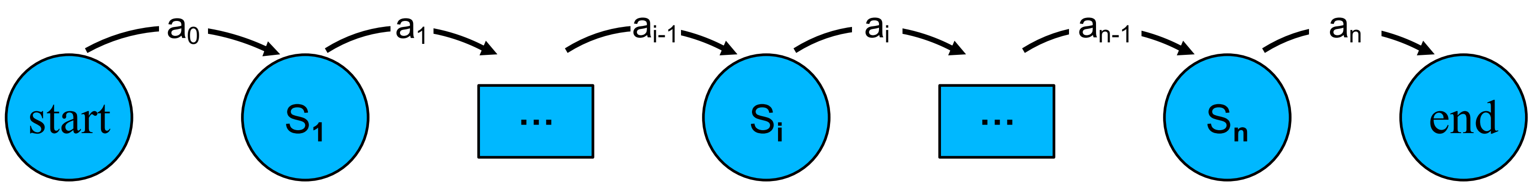
A transition system

- Automata



A transition system

- Automata



Part 3.2: Transition-base Dependency Parsing

Transition-based Dependency Parsing

- Gradually build a tree by applying a sequence of transition actions – shift/reduce (Yamada and Matsumoto, 2003; Nivre, 2003)
- The score of the tree is equal to the summation of the scores of the actions

$$score(X,Y) = \sum_{i=0}^{m} score(X, h_i, a_i)$$

a_i → the action adopted in step i

h_i → the partial results built so far by $a_0 \ldots a_{i-1}$

Y → the tree built by the action sequence $a_0 \ldots a_m$

Transition-based Dependency Parsing

- The goal of a transition-based dependency parser is to find the highest scoring action sequence that builds a legal tree.

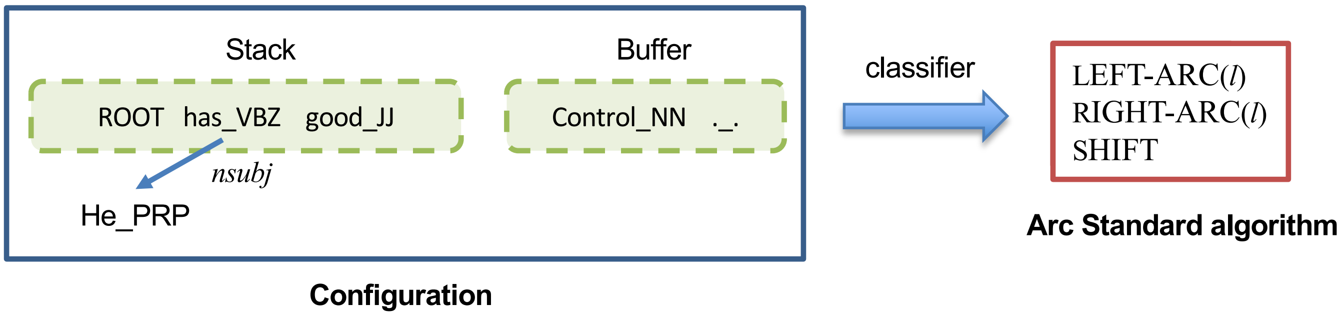
$$Y^* = \arg\max_{Y \in \Phi(X)} score(X, Y)$$

$$= \arg\max_{a_0 \ldots a_m \to Y} \sum_{i=0}^{m} score(X, h_i, a_i)$$

Transition-based Dependency Parsing

- Greedily predict a transition sequence from an initial parser state to some terminal states
- State (configuration)

  = Stack + Buffer + Dependency Arcs



Transition Action: LEFT-ARC (/)

Configuration

Stack	Buffer
ROOT He_PRP has_VBZ	good_JJ Control_NN ._.

Operation:

- Add a left arc (S_0)
- Remove “He_PRP” from Stack

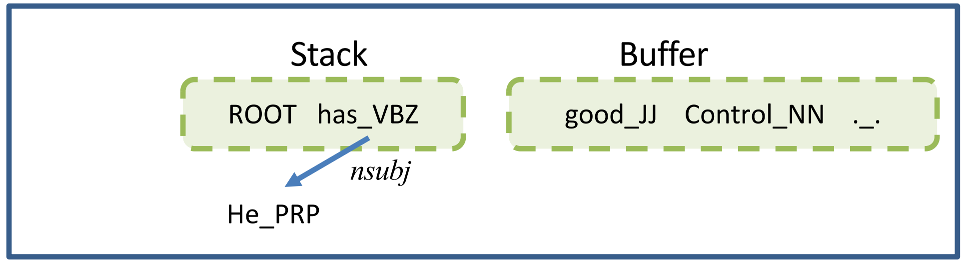
Configuration

Stack	Buffer
ROOT has_VBZ	good_JJ Control_NN ._.

has_VBZ → He_PRP (*nsubj*)

Transition Action: SHIFT

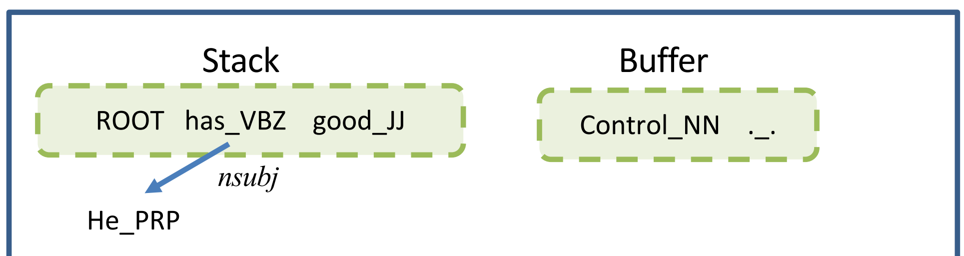
Configuration



Operation:

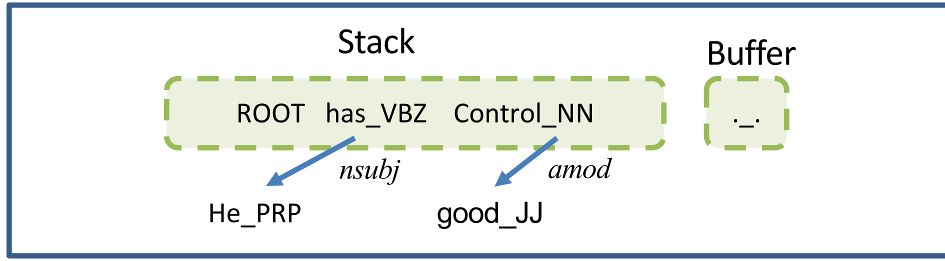
- Shift “good_JJ” from Buffer to top of Stack

Configuration



Transition Action: RIGHT-ARC (/)

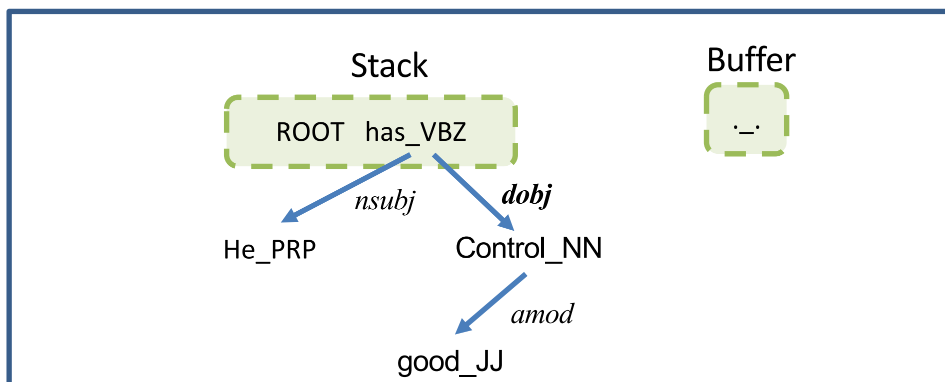
Configuration



Operation:

- Add a right arc (S_1)
- Remove S_0 ("Control_NN") from Stack

Configuration



An Example

Arc-standard Algrithm

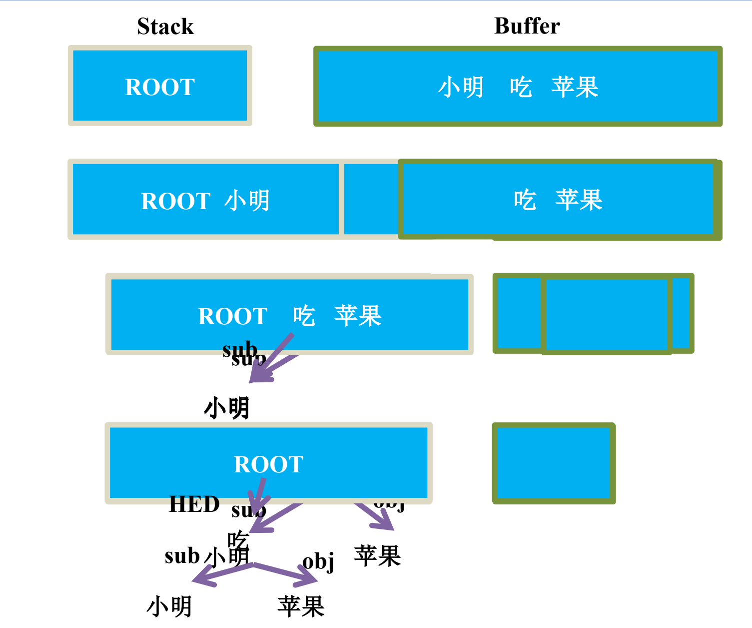
初始状态
Stack只有根节点，待处理词在Buffer中

SHIFT
将Buffer中第一个词压入Stack

LEFT-ARC
弹出Stack中第二个词，生成一条弧从栈顶词指向第二个词

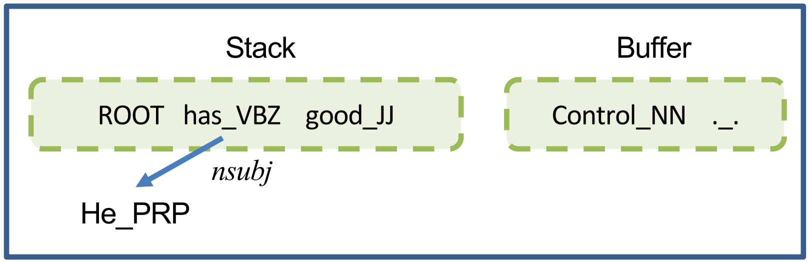
RIGHT-ARC
弹出栈顶词，生成一条弧从栈顶第二个词指向栈顶词

终结状态
Stack只有根节点，Buffer为空

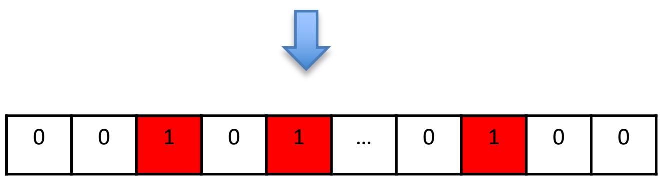


Traditional Features

Configuration



Feature Vector:
- Binary
- Sparse
- High-dimensional



Feature templates: a combination of elements from the configuration.
- For example: (Zhang and Nivre, 2011): 72 feature templates

from single words
S_0wp; S_0w; S_0p; N_0wp; N_0w; N_0p; N_1wp; N_1w; N_1p; N_2wp; N_2w; N_2p;
from word pairs
S_0wpN_0wp; S_0wpN_0w; S_0wN_0wp; S_0wpN_0p; S_0pN_0wp; S_0wN_0w; S_0pN_0p N_0pN_1p
from three words
$N_0pN_1pN_2p$; $S_0pN_0pN_1p$; $S_{0h}pS_0pN_0p$; $S_0pS_{0l}pN_0p$; $S_0pS_{0r}pN_0p$; $S_0pN_0pN_{0l}p$

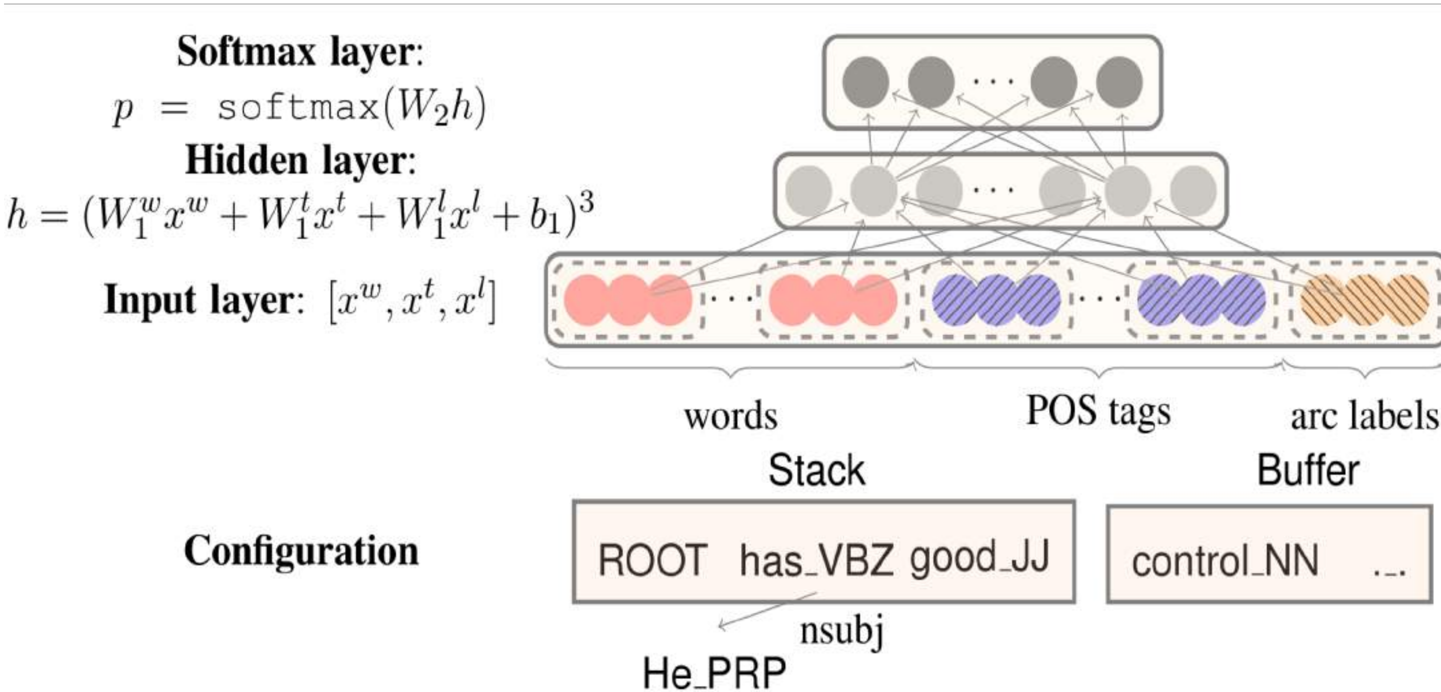
Table 1: Baseline feature templates.
w – word; p – POS-tag.

distance
S_0wd; S_0pd; N_0wd; N_0pd; S_0wN_0wd; S_0pN_0pd;
valency
S_0wv_r; S_0pv_r; S_0wv_l; S_0pv_l; N_0wv_l; N_0pv_l;
unigrams
$S_{0h}w$; $S_{0h}p$; S_0l; $S_{0l}w$; $S_{0l}p$; $S_{0l}l$; $S_{0r}w$; $S_{0r}p$; $S_{0r}l$;$N_{0l}w$; $N_{0l}p$; $N_{0l}l$;
third-order
$S_{0h2}w$; $S_{0h2}p$; $S_{0h}l$; $S_{0l2}w$; $S_{0l2}p$; $S_{0l2}l$; $S_{0r2}w$; $S_{0r2}p$; $S_{0r2}l$; $N_{0l2}w$; $N_{0l2}p$; $N_{0l2}l$; $S_0pS_{0l}pS_{0l2}p$; $S_0pS_{0r}pS_{0r2}p$; $S_0pS_{0h}pS_{0h2}p$; $N_0pN_{0l}pN_{0l2}p$;
label set
S_0ws_r; S_0ps_r; S_0ws_l; S_0ps_l; N_0ws_l; N_0ps_l;

Table 2: New feature templates.
w – word; p – POS-tag; v_l, v_r – valency; l – dependency label, s_l, s_r – labelset.

Part 3.2: Neural Transition-based Dependency Parsing

Neural Action Classifier



Chen, D., & Manning, C. D. (2014). A Fast and Accurate Dependency Parser using Neural Network. ACL.

Results

Parser	Dev		Test		Speed
	UAS	LAS	UAS	LAS	(sent/s)
standard	90.2	87.8	89.4	87.3	26
eager	89.8	87.4	89.6	87.4	34
Malt:sp	89.8	87.2	89.3	86.9	469
Malt:eager	89.6	86.9	89.4	86.8	448
MSTParser	91.4	88.1	90.7	87.6	10
Our parser	**92.0**	**89.7**	**91.8**	**89.6**	**654**

PTB (SD)

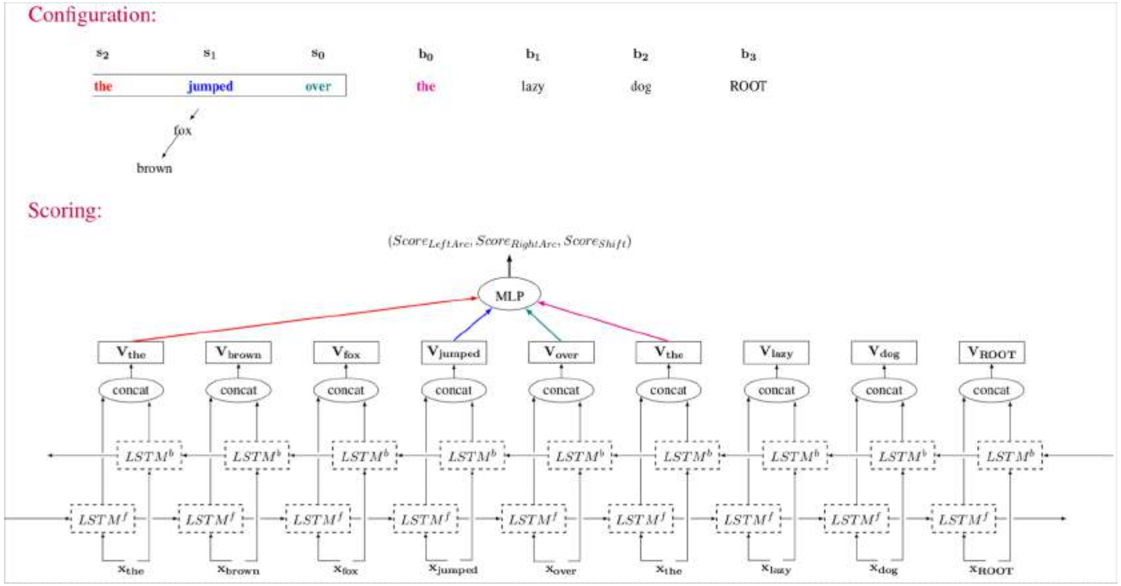
Parser	Dev		Test		Speed
	UAS	LAS	UAS	LAS	(sent/s)
standard	82.4	80.9	82.7	81.2	72
eager	81.1	79.7	80.3	78.7	80
Malt:sp	82.4	80.5	82.4	80.6	420
Malt:eager	81.2	79.3	80.2	78.4	393
MSTParser	**84.0**	82.1	83.0	81.2	6
Our parser	**84.0**	**82.4**	**83.9**	**82.4**	**936**

CTB (SD)

Chen, D., & Manning, C. D. (2014). A Fast and Accurate Dependency Parser using Neural Network. ACL.

LSTM Feature Extractor

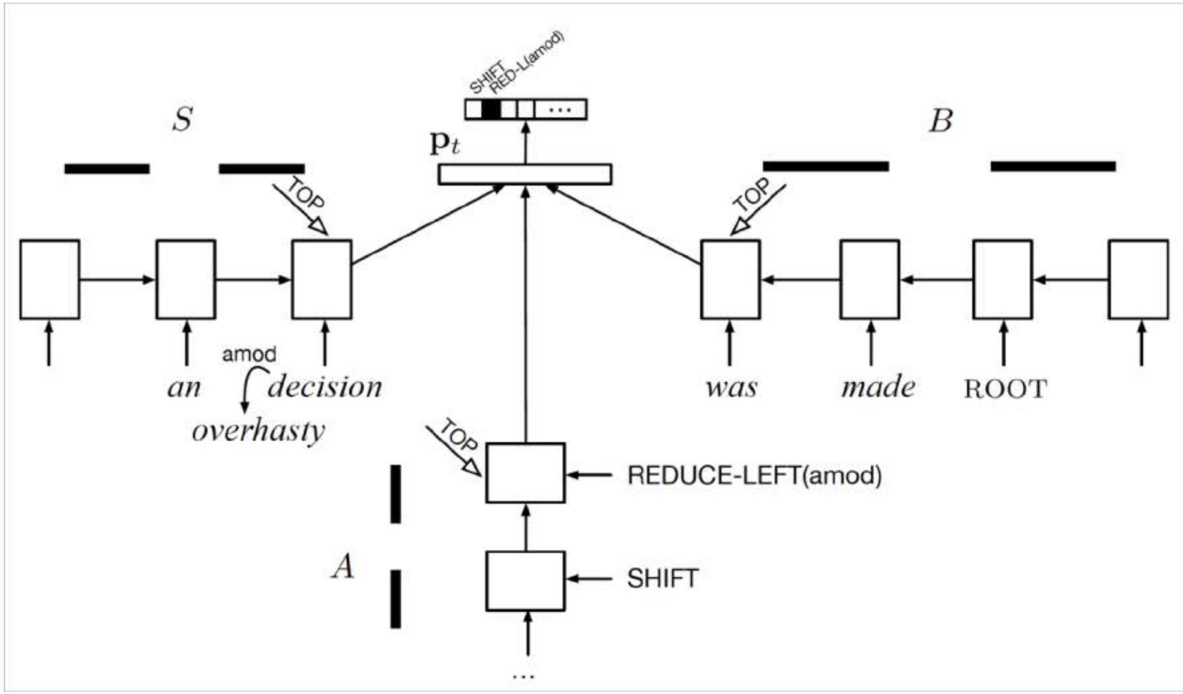
- Chen and Manning with richer (LSTM) features



Keperwasser, E., & Goldberg, Y. (2016). Simple and Accurate Dependency Parsing Using Bidirectional LSTM Feature Representations. TACL.

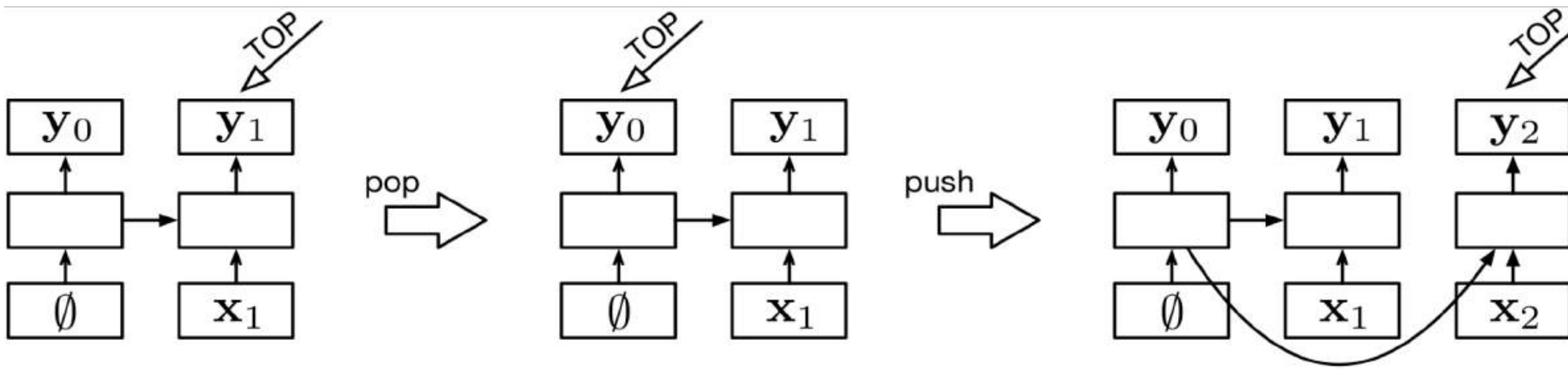
Stack LSTM

- Dyer Parser (Chen and Manning with less features)



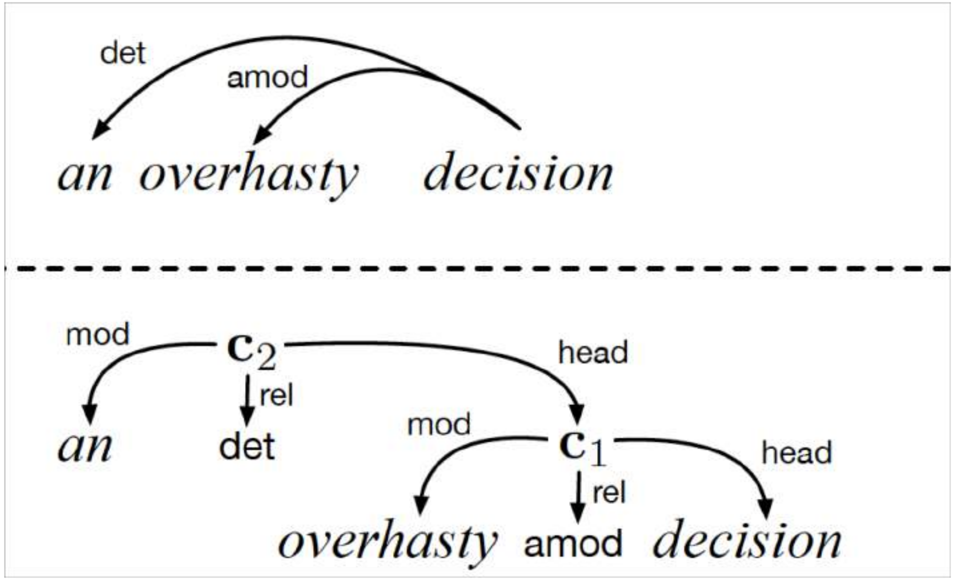
Dyer, C., Ballesteros, M., Ling, W., Matthews, A., & Smith, N. A. (2015). Transition-Based Dependency Parsing with Stack Long Short-Term Memory. ACL.

Stack LSTM



Dyer, C., Ballesteros, M., Ling, W., Matthews, A., & Smith, N. A. (2015). Transition-Based Dependency Parsing with Stack Long Short-Term Memory. ACL.

Subtree Representation (Recursive NN)



Dyer, C., Ballesteros, M., Ling, W., Matthews, A., & Smith, N. A. (2015). Transition-Based Dependency Parsing with Stack Long Short-Term Memory. ACL.

Results

	Development		Test	
	UAS	LAS	UAS	LAS
S-LSTM	**93.2**	**90.9**	**93.1**	**90.9**
−POS	93.1	90.4	92.7	90.3
−pretraining	92.7	90.4	92.4	90.0
−composition	92.7	89.9	92.2	89.6
S-RNN	92.8	90.4	92.3	90.1
C&M (2014)	92.2	89.7	91.8	89.6

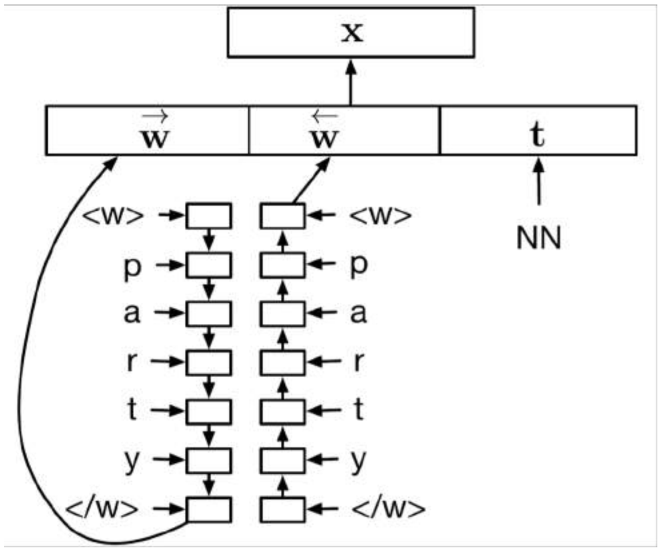
PTB (SD)

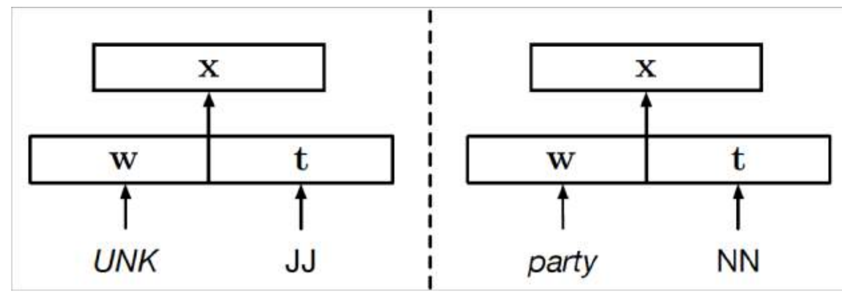
	Dev. set		Test set	
	UAS	LAS	UAS	LAS
S-LSTM	**87.2**	**85.9**	**87.2**	**85.7**
−composition	85.8	84.0	85.3	83.6
−pretraining	86.3	84.7	85.7	84.1
−POS	82.8	79.8	82.2	79.1
S-RNN	86.3	84.7	86.1	84.6
C&M (2014)	84.0	82.4	83.9	82.4

CTB (CTB5)

Dyer, C., Ballesteros, M., Ling, W., Matthews, A., & Smith, N. A. (2015). Transition-Based Dependency Parsing with Stack Long Short-Term Memory. ACL.

Character based Word Vector



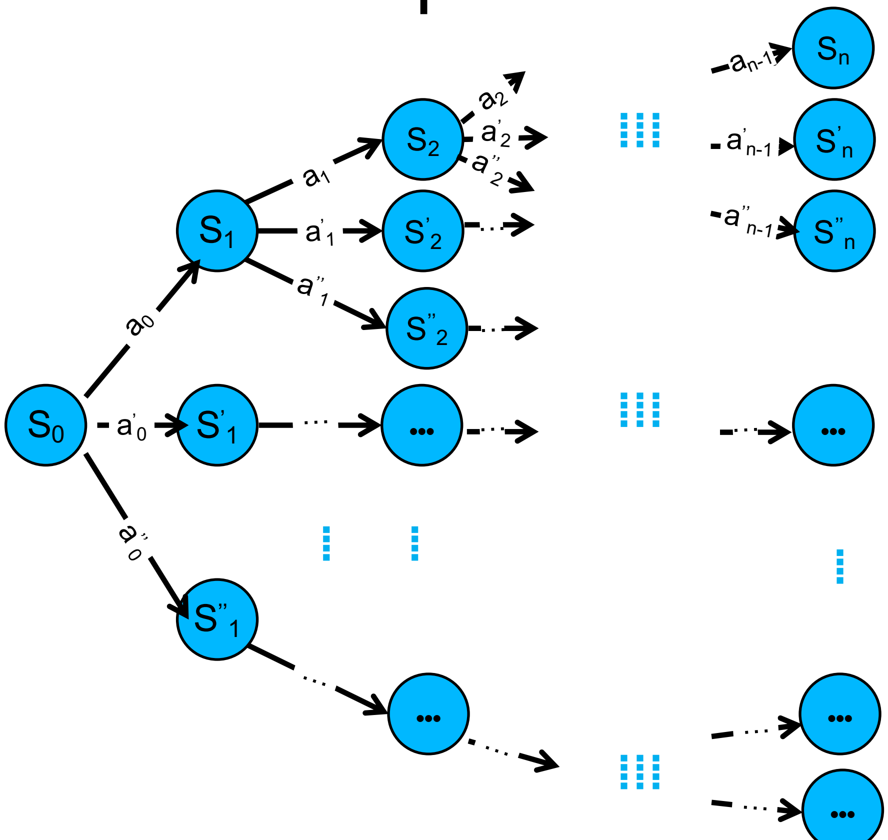


Ballesteros, M., Dyer, C., & Smith, N. A. (2015). Improved Transition-Based Parsing by Modeling Characters instead of Words with LSTMs. EMNLP.

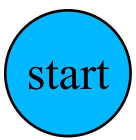
Part 3.3: Transition-base Methods with Beam-search Decoding

Search

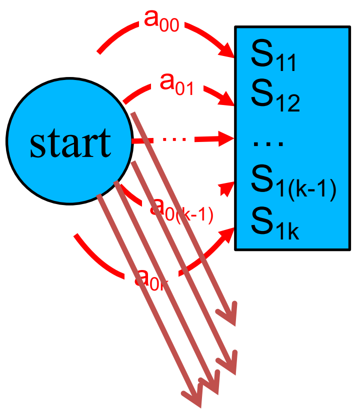
- Find the best sequence of actions



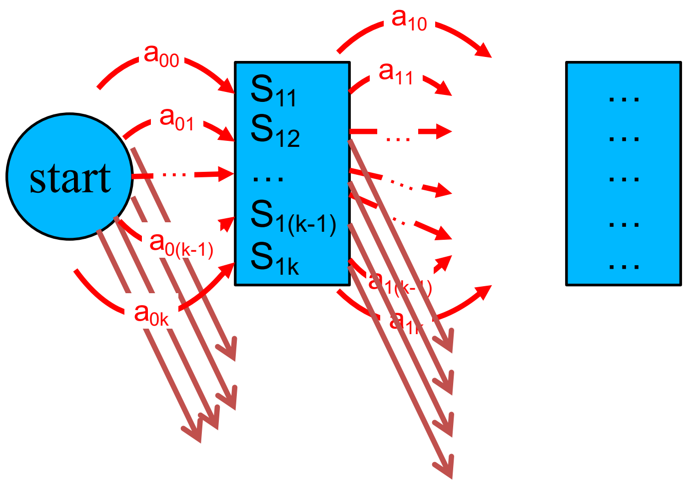
Beam-search decoding



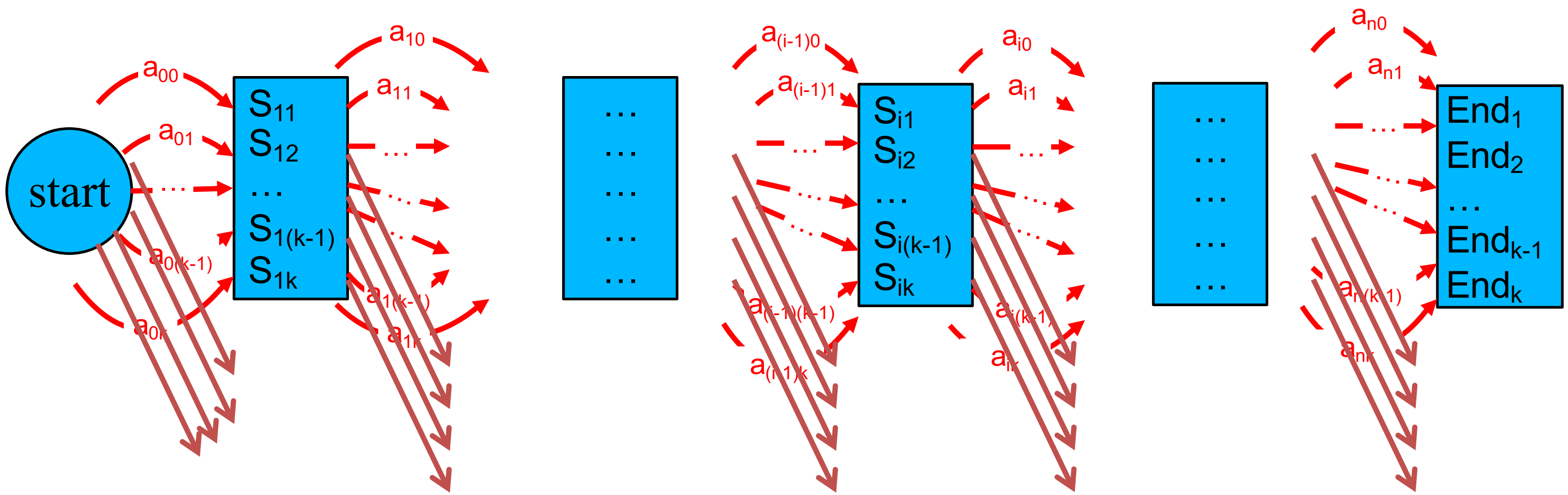
Beam-search decoding



Beam-search decoding



Beam-search decoding



Sentence-level Log Likelihood

$$p(y_i \mid x, \theta) = \frac{e^{f(x,\, \theta)_i}}{\sum_{y_j \in \mathrm{GEN}(x)} e^{f(x,\, \theta)_j}}$$

$$f(x,\, \theta)_i = \sum_{a_k \in y_i} o(x,\, y_i,\, k,\, a_k)$$

Zhou, H., Zhang, Y., Huang, S., & Chen, J. (2015). A Neural Probabilistic Structured-Prediction Model for Transition-Based Dependency Parsing. ACL.

Contrastive Estimation

$$L(\theta) = - \sum_{(x_i, y_i) \in (X,Y)} \log p(y_i \mid x_i, \theta)$$

$$= - \sum_{(x_i, y_i) \in (X,Y)} \log \frac{e^{f(x_i, \theta)_i}}{Z(x_i, \theta)}$$

$$= \sum_{(x_i, y_i) \in (X,Y)} \log Z(x_i, \theta) - f(x_i, \theta)_i$$

$$Z(x, \theta) = \sum_{y_j \in \mathrm{GEN}(x)} e^{f(x, \theta)_j}$$

Zhou, H., Zhang, Y., Huang, S., & Chen, J. (2015). A Neural Probabilistic Structured-Prediction Model for Transition-Based Dependency Parsing. ACL.

Contrastive Estimation

$$L'(\theta) = -\sum_{(x_i, y_i) \in (X,Y)} \log p'(y_i \mid x_i, \theta)$$

$$= -\sum_{(x_i, y_i) \in (X,Y)} \log \frac{e^{f(x_i,\theta)_i}}{Z'(x_i, \theta)}$$

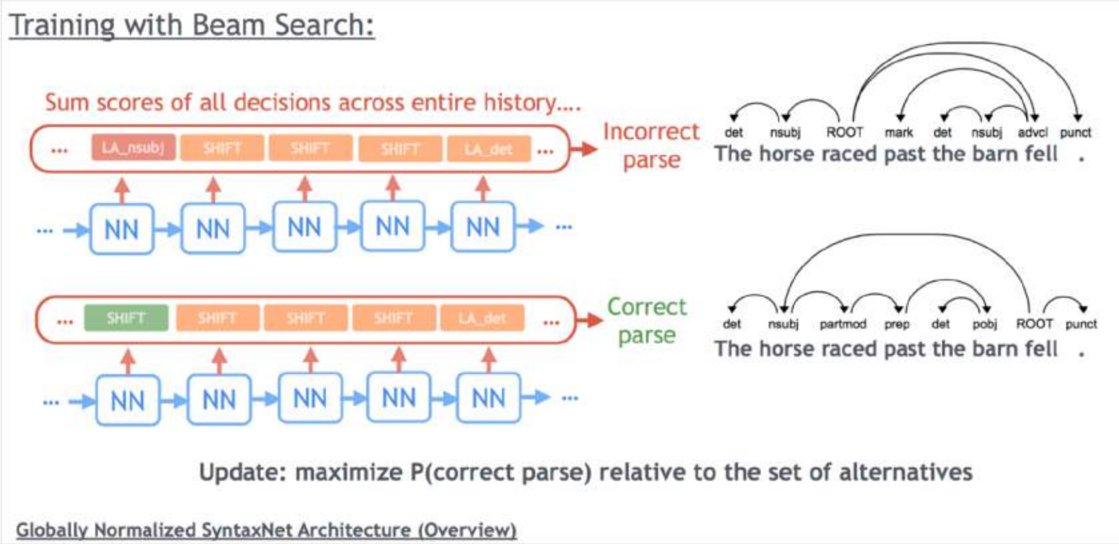
$$= \sum_{(x_i, y_i) \in (X,Y)} \log Z'(x_i, \theta) - f(x_i, \theta)_i$$

$$Z'(x, \theta) = \sum_{y_j \in \mathrm{BEAM}(x)} e^{f(x, \theta)_j}$$

Zhou, H., Zhang, Y., Huang, S., & Chen, J. (2015). A Neural Probabilistic Structured-Prediction Model for Transition-Based Dependency Parsing. ACL.

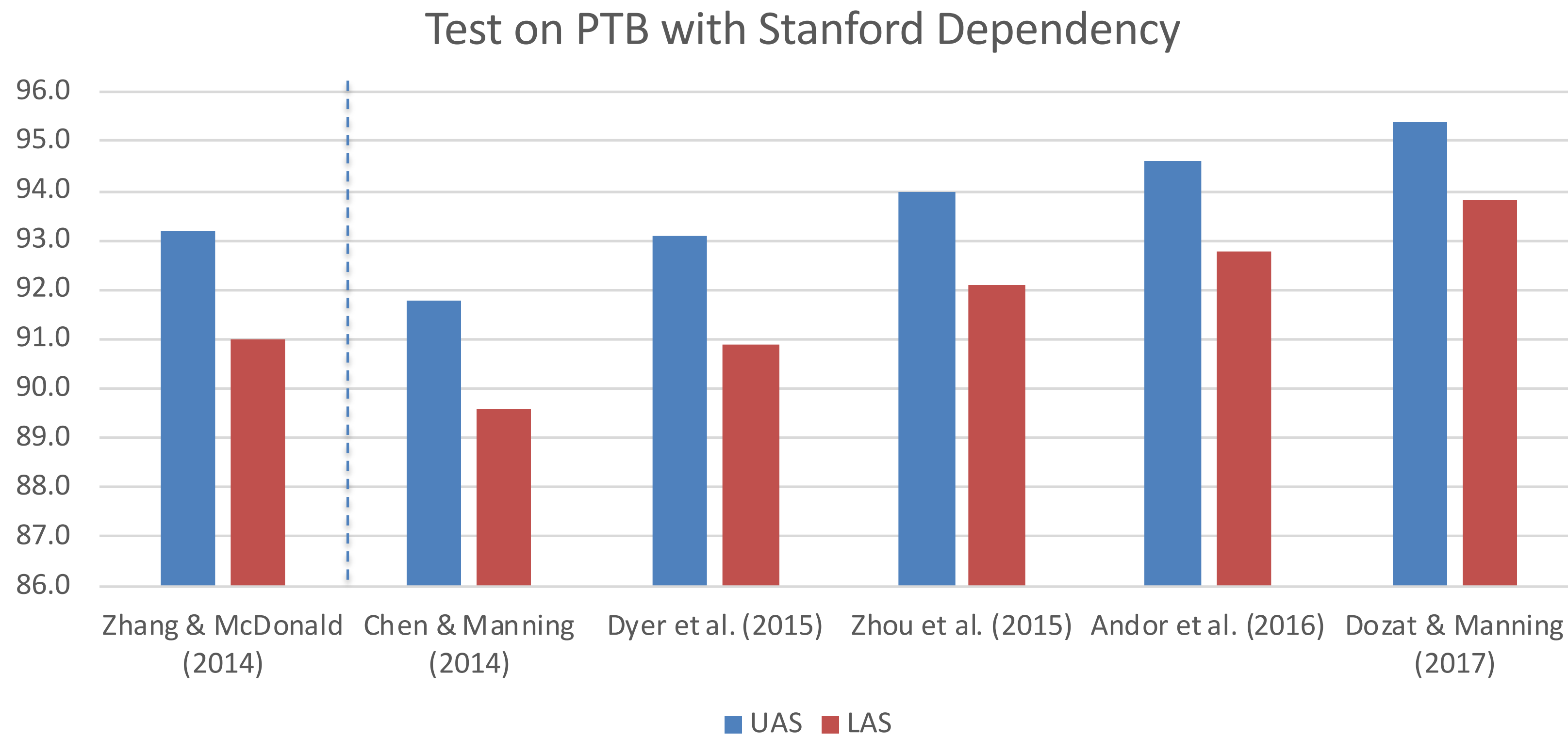
Google's SyntaxNet

- Andor et al. follows this method
  - Offers theorem
  - Tries more tasks
  - Get better results



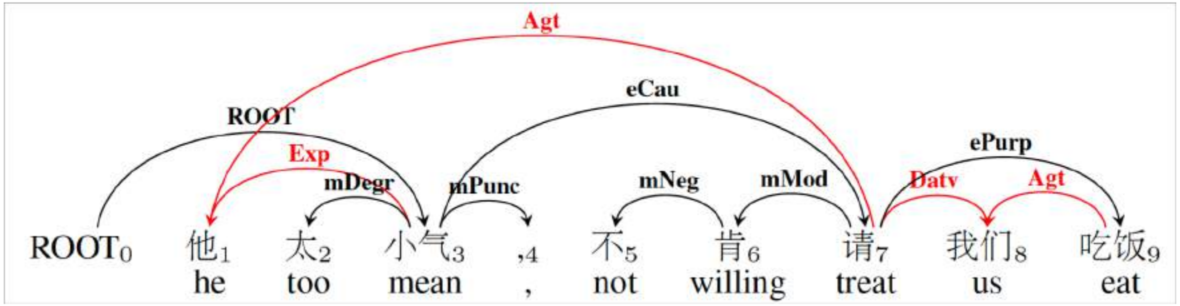
Andor, D., Alberti, Chris., Weiss, D., Severyn, A., Presta, A., Ganchev, K., Petrov, S., & Collins, M. (2016). Globally Normalized Transition-Based Neural Networks. ACL.

Changes of Performance



Part 3.4: Advanced Topics

Semantic Dependency Graph



		Train	Dev	Test
NEWS	#sent	8,301	534	1,233
	#word	250,249	15,325	34,305
TEXT	#sent	10,817	1,546	3,096
	#word	128,095	18,257	36,097

Wanxiang Che, Yu Ding, Yanqiu Shao, Ting Liu. **SemEval-2016 Task 9**: Chinese Semantic Dependency Parsing.

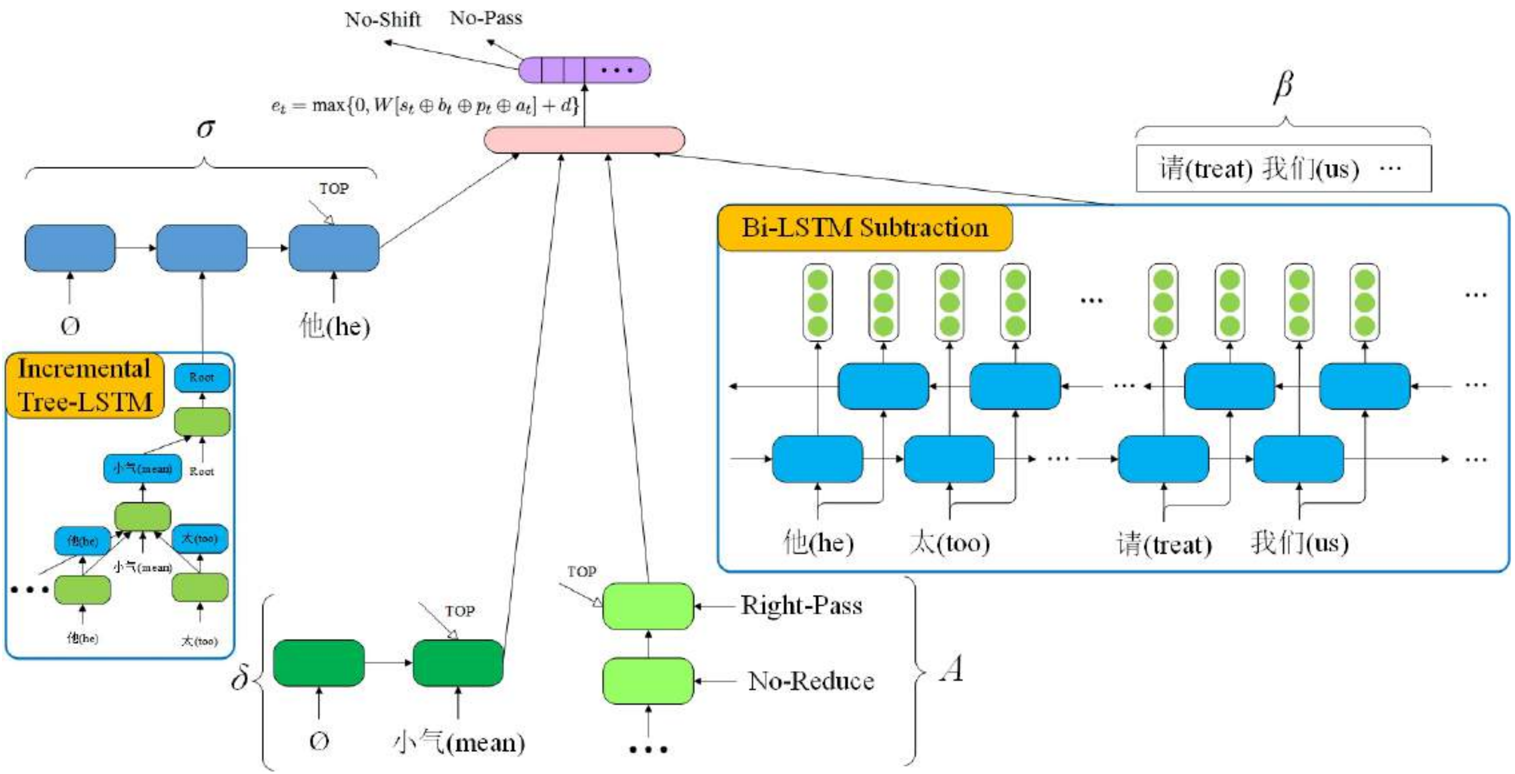
Semantic Dependency Graph

- List-based transition system

- New transition actions
  - Left-Reduce, Right-Shift, No-Shift, No-Reduce, Left-Pass, Right-Pass, No-Pass

Yuxuan Wang, Jiang Guo, Wanxiang Che and Ting Liu. Transition-Based Chinese Semantic Dependency Graph Parsing. CCL 2016. **Best Paper Award.**

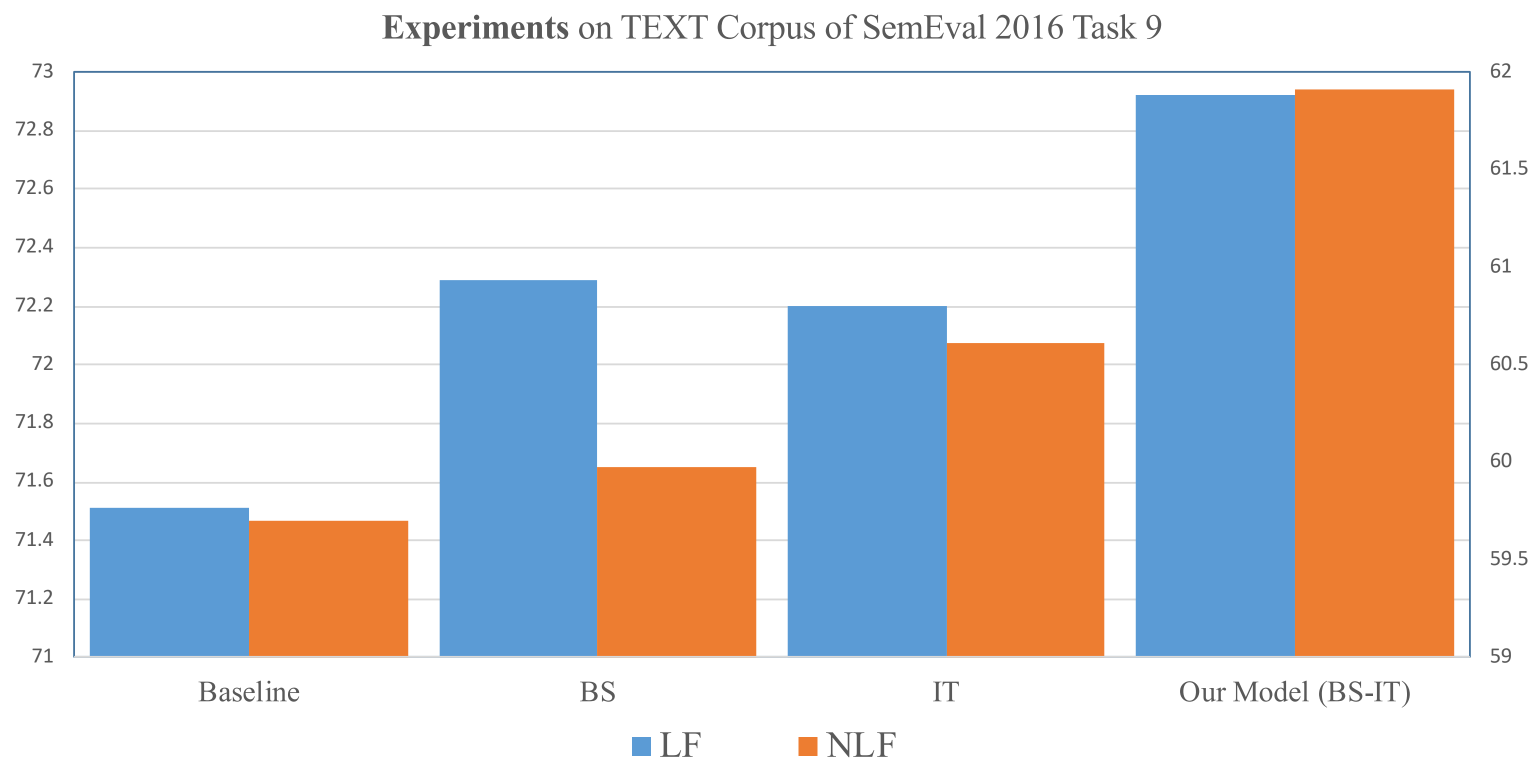
IT-BS Classifier



Yuxuan Wang, Wanxiang Che, Jiang Guo and Ting Liu. A Neural Transition-Based Approach for Semantic Dependency Graph Parsing. AAAI 2018.

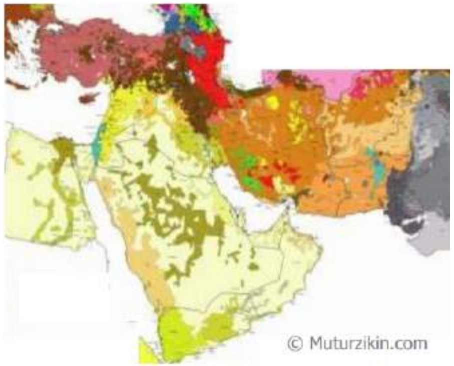
Semantic Dependency Graph

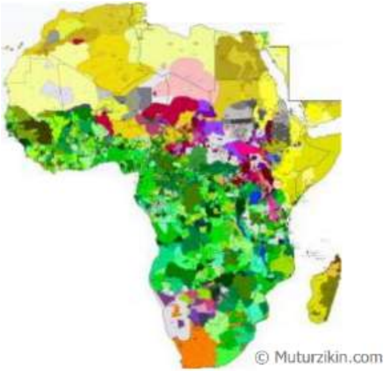
Results



Multilingual Dependency Parsing

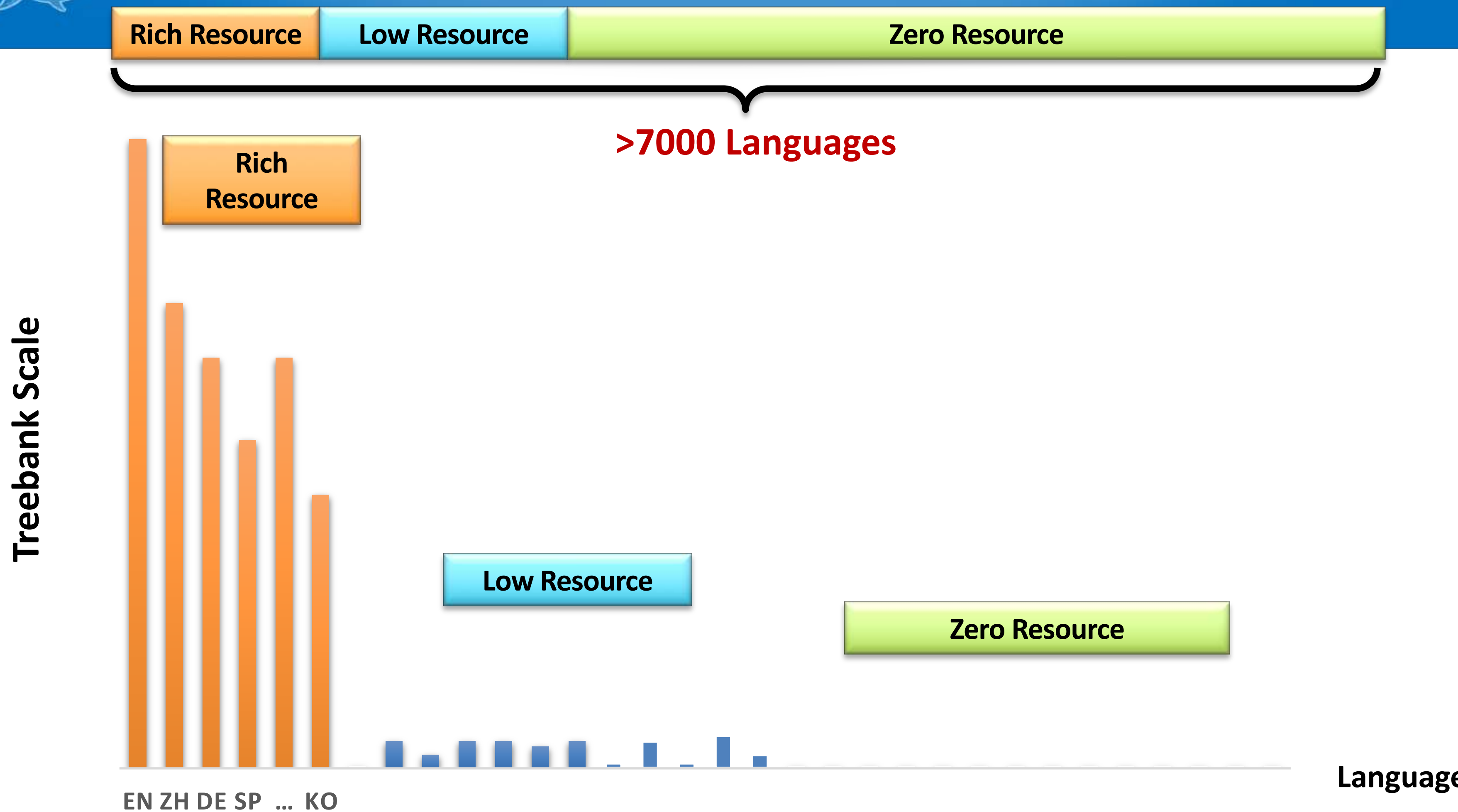
- Over 7,000 languages all around the world
  - Most of the languages are ***low-resource*** for dependency parsing



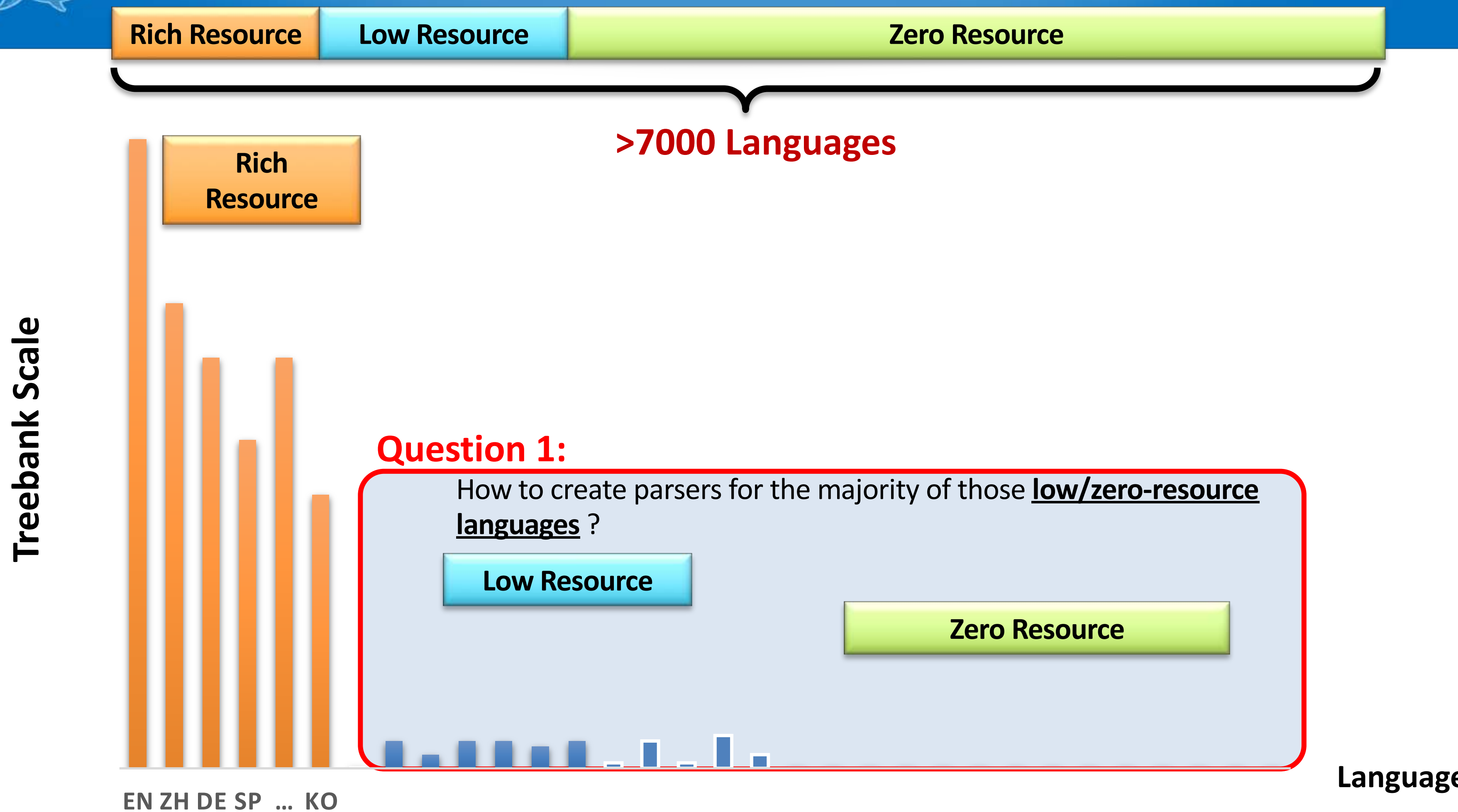


(Colors indicate language families)

≈ 30 languages

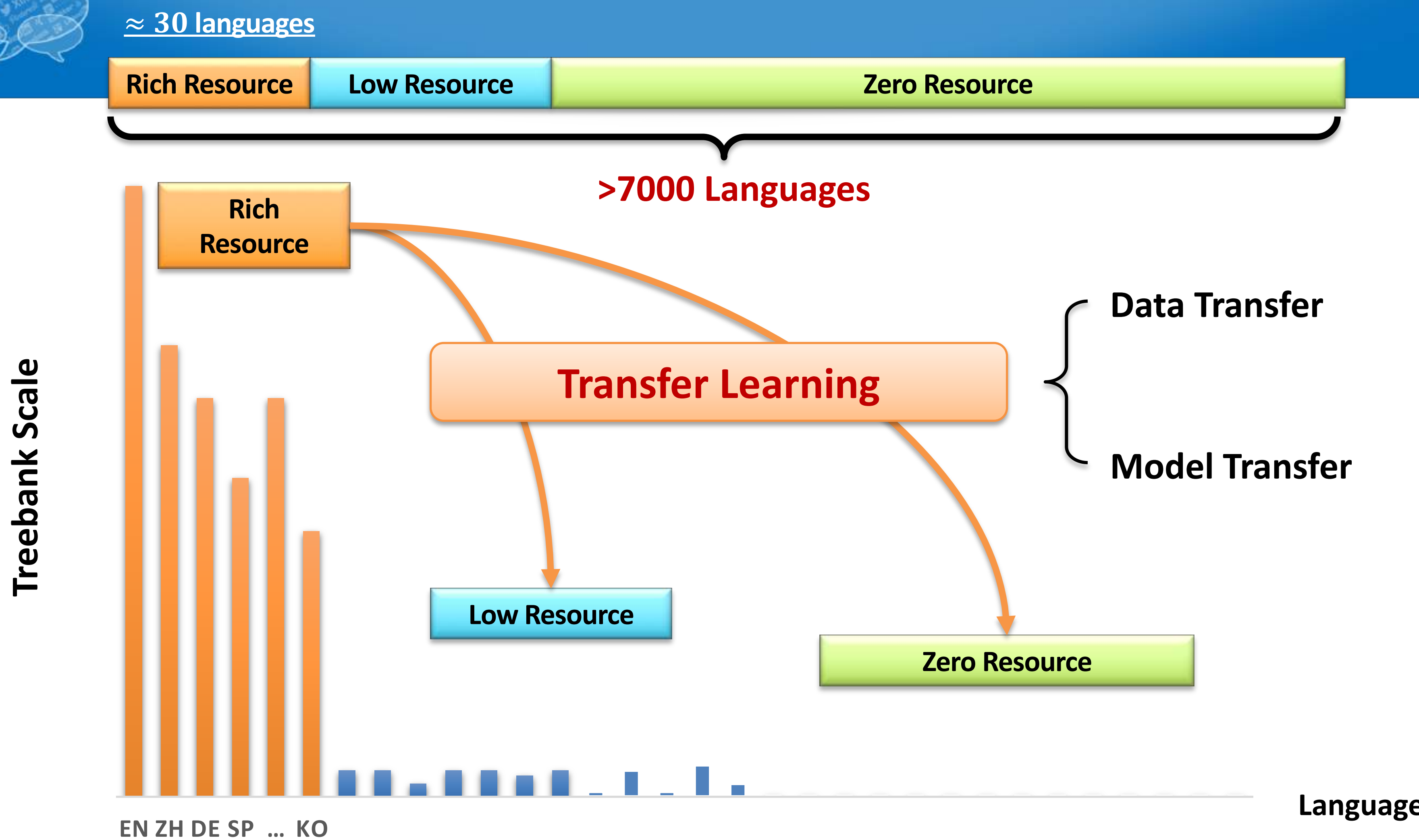


≈ 30 languages



Question 1:

How to create parsers for the majority of those **low/zero-resource languages** ?



≈ 30 languages
Rich Resource
Low Resource
Zero Resource
>7000 Languages
Rich Resource
Transfer Learning
Data Transfer
Model Transfer
Low Resource
Zero Resource
Treebank Scale
Language
EN ZH DE SP … KO

≈ 30 languages

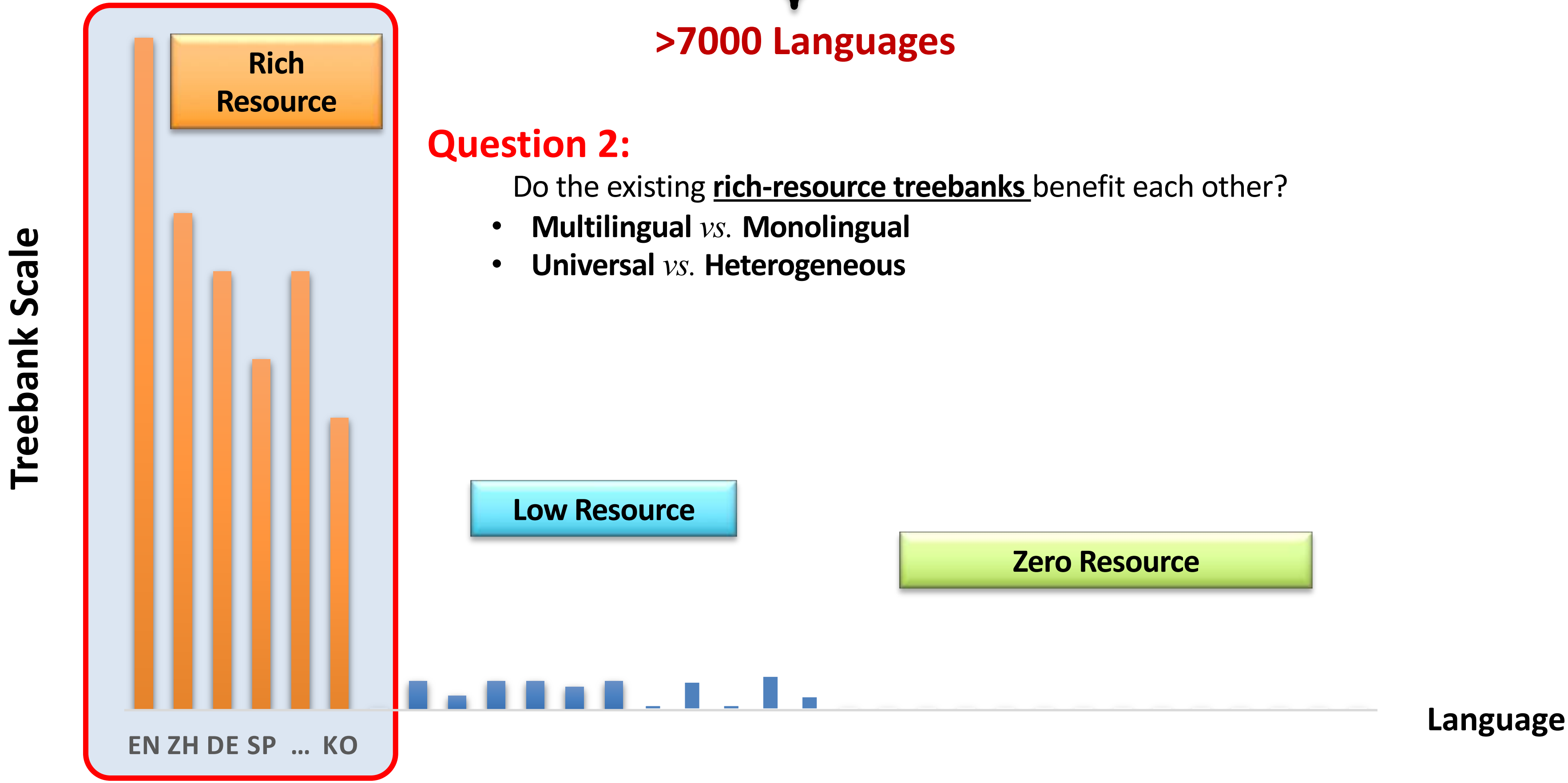
Rich Resource	Low Resource	Zero Resource

>7000 Languages

Question 2:

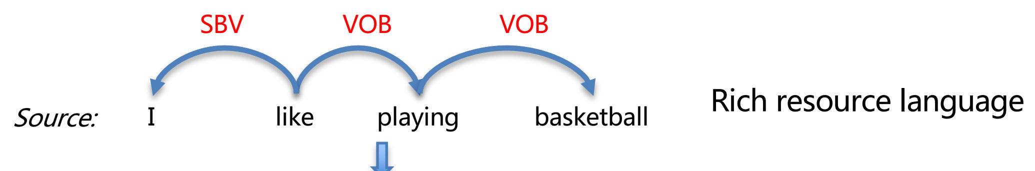
Do the existing **rich-resource treebanks** benefit each other?

- **Multilingual** *vs.* **Monolingual**
- **Universal** *vs.* **Heterogeneous**

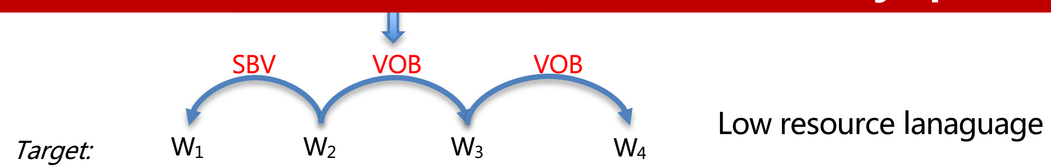


Cross-lingual Dependency Parsing

- Use the model trained on source language to parse the target language

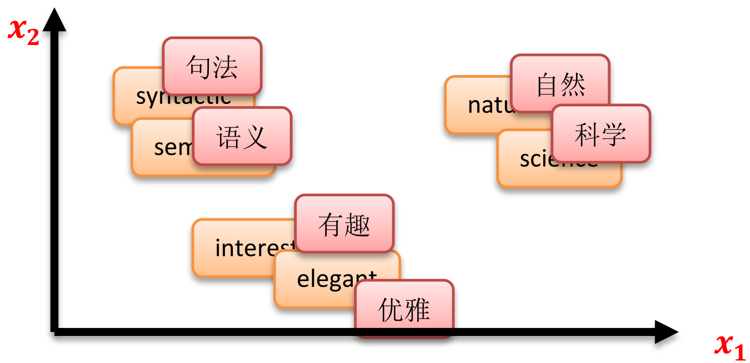


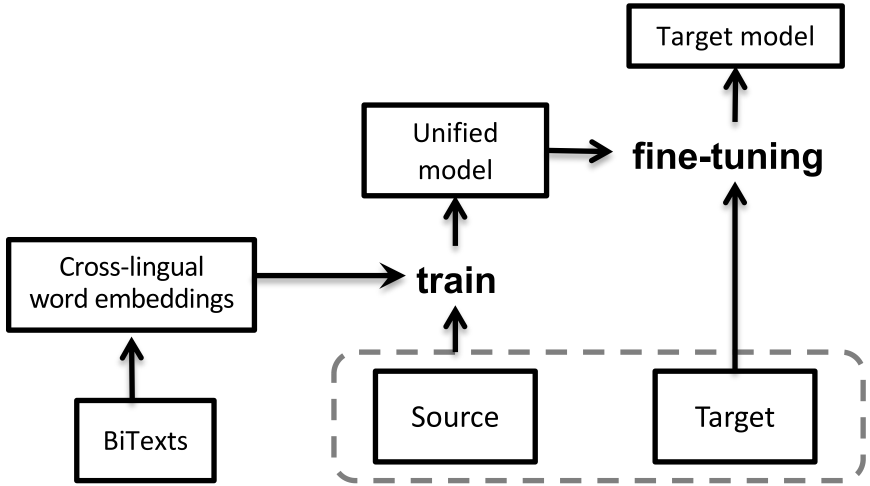
How to overcome the lexical inconsistency problem?



Cross-lingual Dependency Parsing

- Use bi-lingual word embeddings to overcome the lexical inconsistency problem



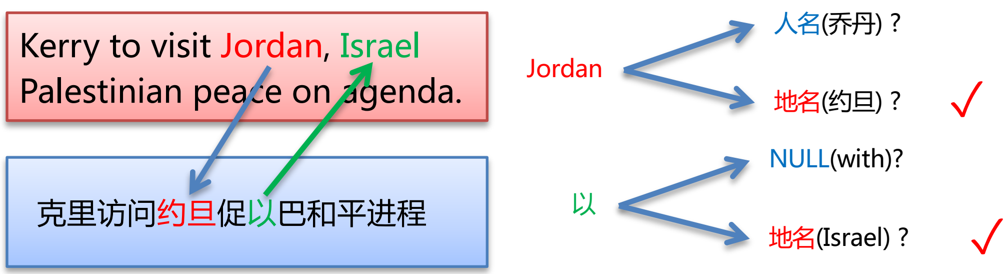


- The performance of target language can be improved more than 4%

Our paper: ACL 2015， AAAI 2016，JAIR 2016，CoNLL 2017

Bi-lingual based Named Entity Recognition

- The parallel corpus have inter-translated named entities

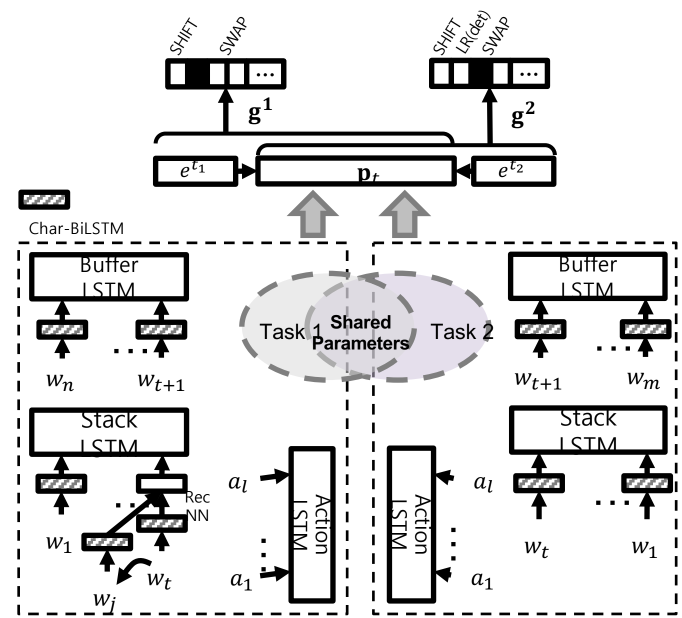


- Bi-lingual constraint based methods

Our paper: NAACL 2013、ACL 2013、AAAI 2013 (Outstanding mention award)

Deep Multi-task Learning Framework

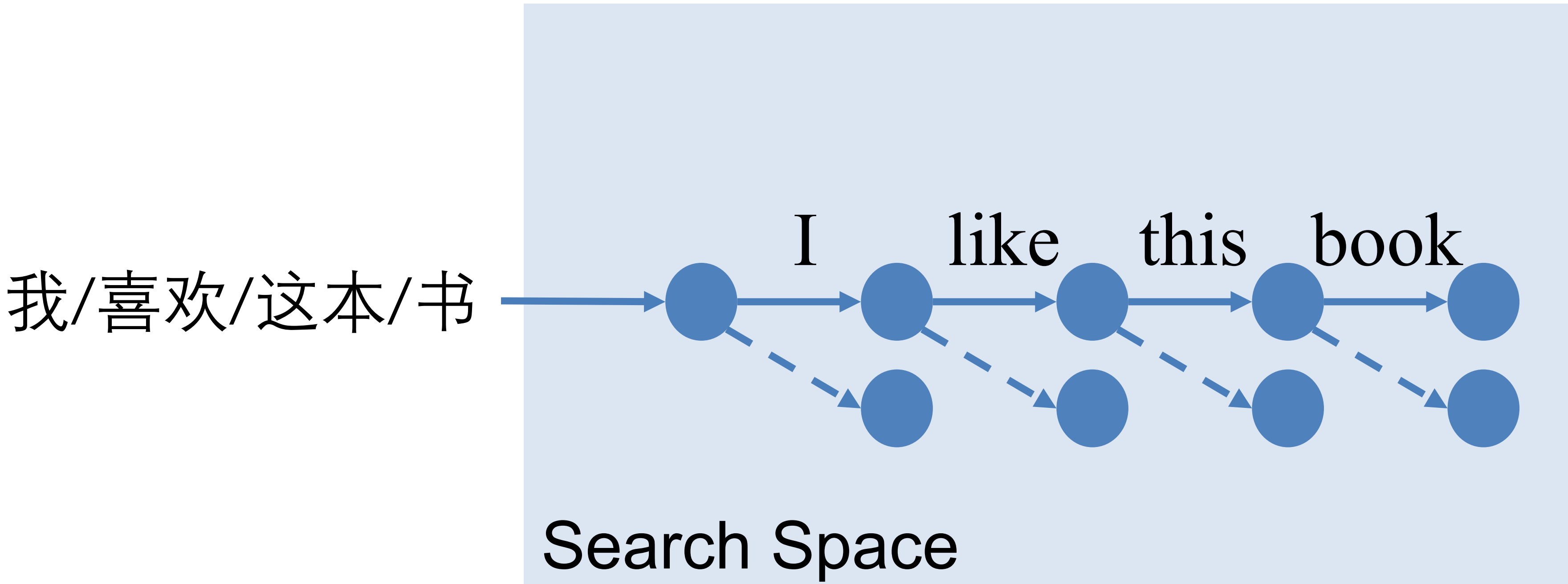
- Each corpus can be looked as a task
  - Multi-lingual treebanks
  - Mono-lingual heterogeneous treebanks
  - Multiple NLP tasks
- **Shared parameters**
  - LSTM(B), LSTM(S)
  - LSTM(A)
  - BiLSTM(chars)
  - RecNN
  - W_A, W_B, W_S
  - $E_{pos}, E_{char}, E_{rel}, E_{act}$



Jiang Guo, Wanxiang Che, Haifeng Wang and Ting Liu. A Universal Framework for Transfer Parsing across Multi-typed Treebanks. Coling 2016.

Distilling Knowledge for Transition-based Structured Prediction

Transition-based Machine Translation

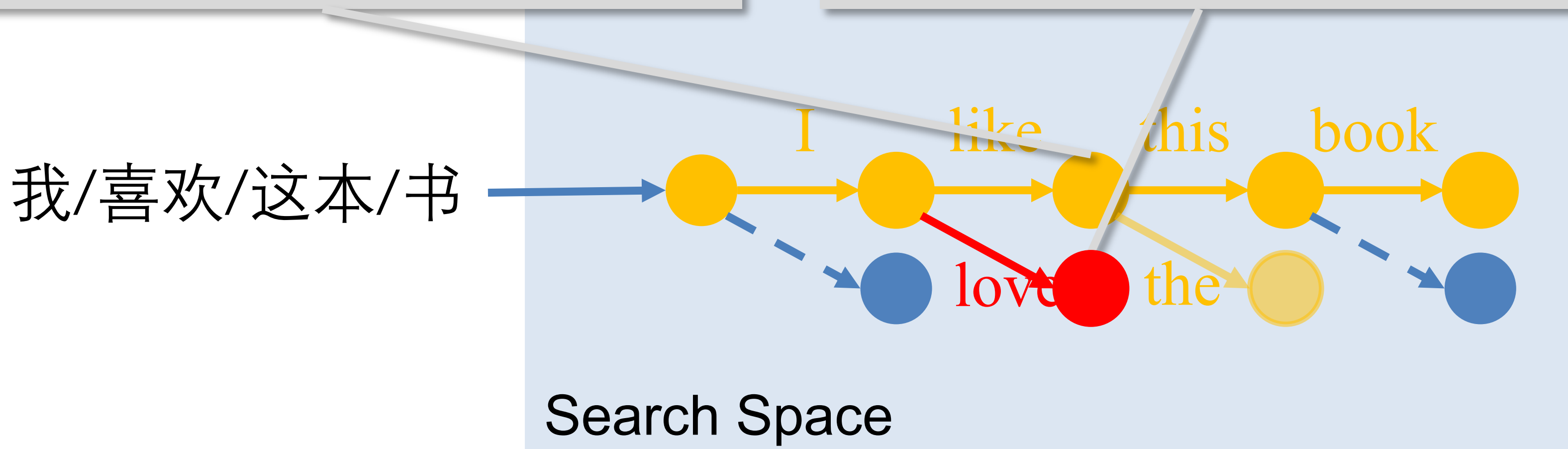


Yijia Liu, Wanxiang Che, Huaipeng Zhao, Bing Qin and Ting Liu. Distilling Knowledge for Search-based Structured Prediction. ACL 2018.

Problems of the Generic Learning Algorithm

Ambiguities in training data
"both *this* and *the* seems reasonable"

Training and test discrepancy
"What if I made wrong decision?"

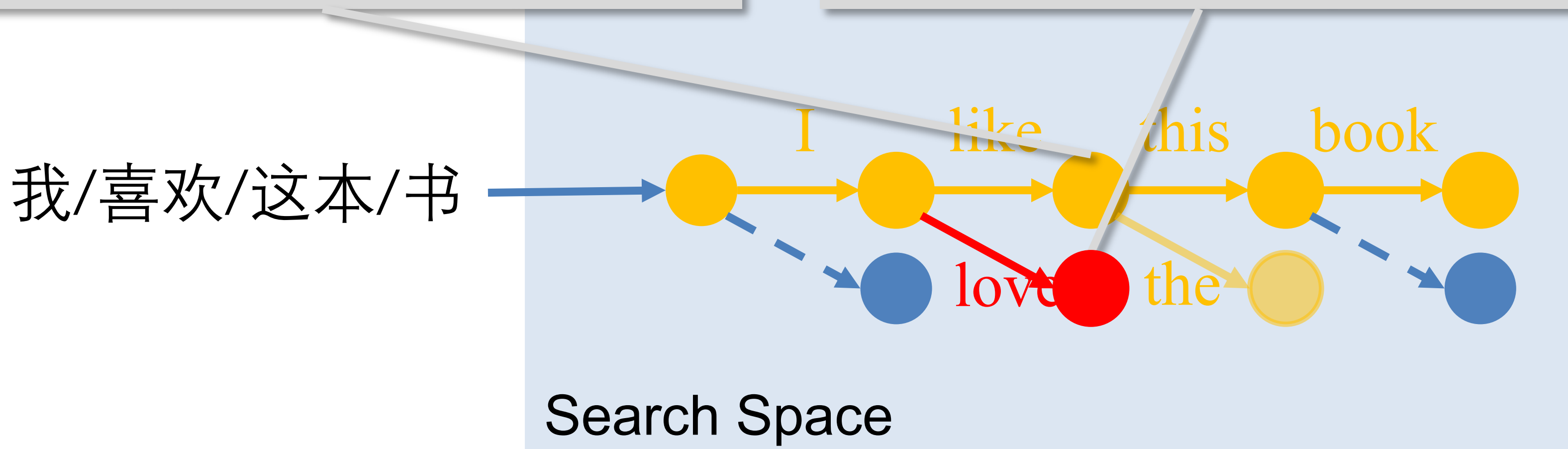


Yijia Liu, Wanxiang Che, Huaipeng Zhao, Bing Qin and Ting Liu. Distilling Knowledge for Search-based Structured Prediction. ACL 2018.

Problems of the Generic Learning Algorithm

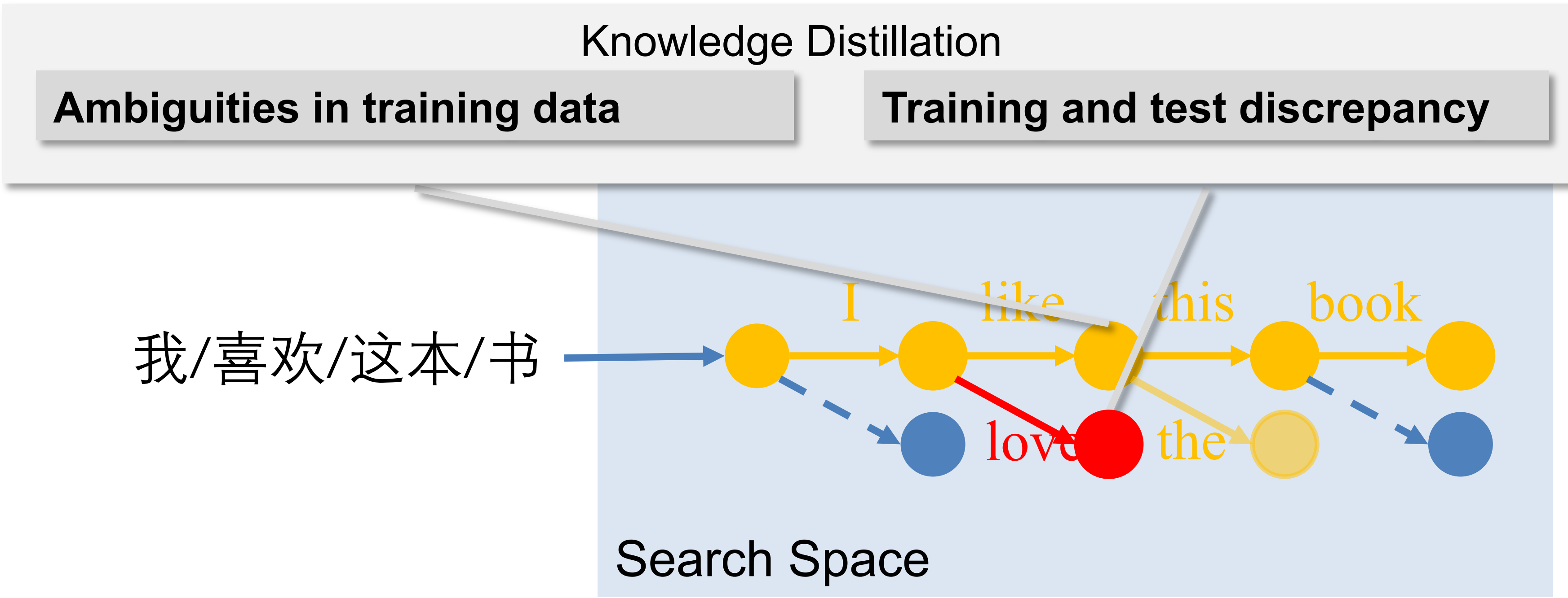
Ambiguities in training data
Ensemble *(Dietterich, 2000)*

Training and test discrepancy
Explore *(Ross and Bagnell, 2010)*



Yijia Liu, Wanxiang Che, Huaipeng Zhao, Bing Qin and Ting Liu. Distilling Knowledge for Search-based Structured Prediction. ACL 2018.

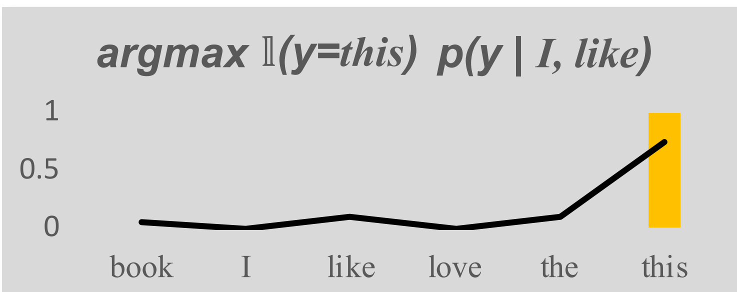
Problems of the Generic Learning Algorithm



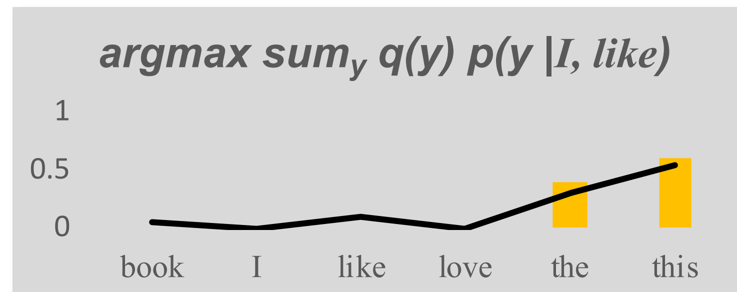
Yijia Liu, Wanxiang Che, Huaipeng Zhao, Bing Qin and Ting Liu. Distilling Knowledge for Search-based Structured Prediction. ACL 2018.

Knowledge Distillation

Learning from negative log-likelihood



Learning from knowledge distillation

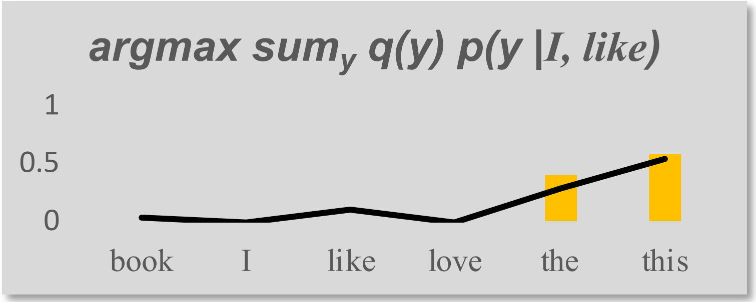


$q(y \mid I, like)$ is the output distribution of a ***teacher*** model (e.g. ensemble)

Yijia Liu, Wanxiang Che, Huaipeng Zhao, Bing Qin and Ting Liu. Distilling Knowledge for Search-based Structured Prediction. ACL 2018.

Knowledge Distillation: from Where

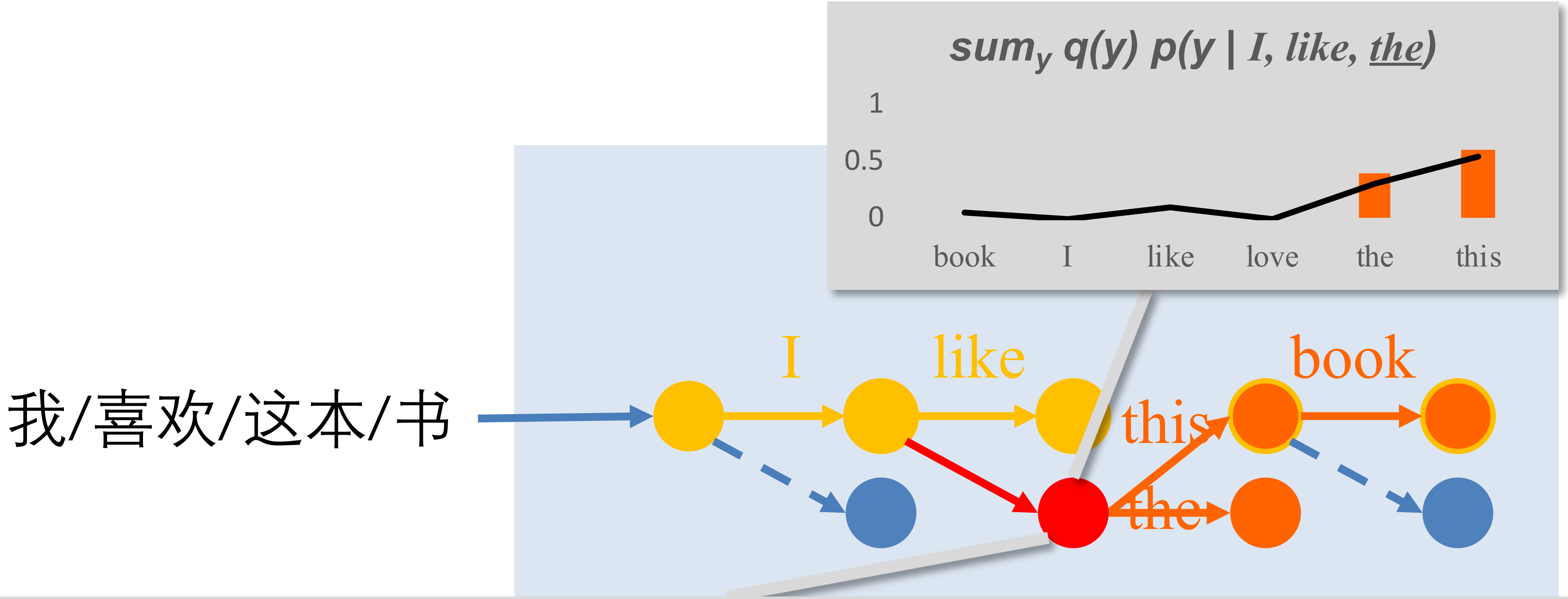
Learning from knowledge distillation



Ambiguities in training data
Ensemble *(Dietterich, 2000)*
We use ensemble of M structure predictor as the ***teacher q***

Yijia Liu, Wanxiang Che, Huaipeng Zhao, Bing Qin and Ting Liu. Distilling Knowledge for Search-based Structured Prediction. ACL 2018.

KD for Transition-based Structured Prediction on Explored Data



Training and test discrepancy

Explore *(Ross and Bagnell, 2010)*

We use ***teacher q*** to explore the search space & learn from KD on the explored data

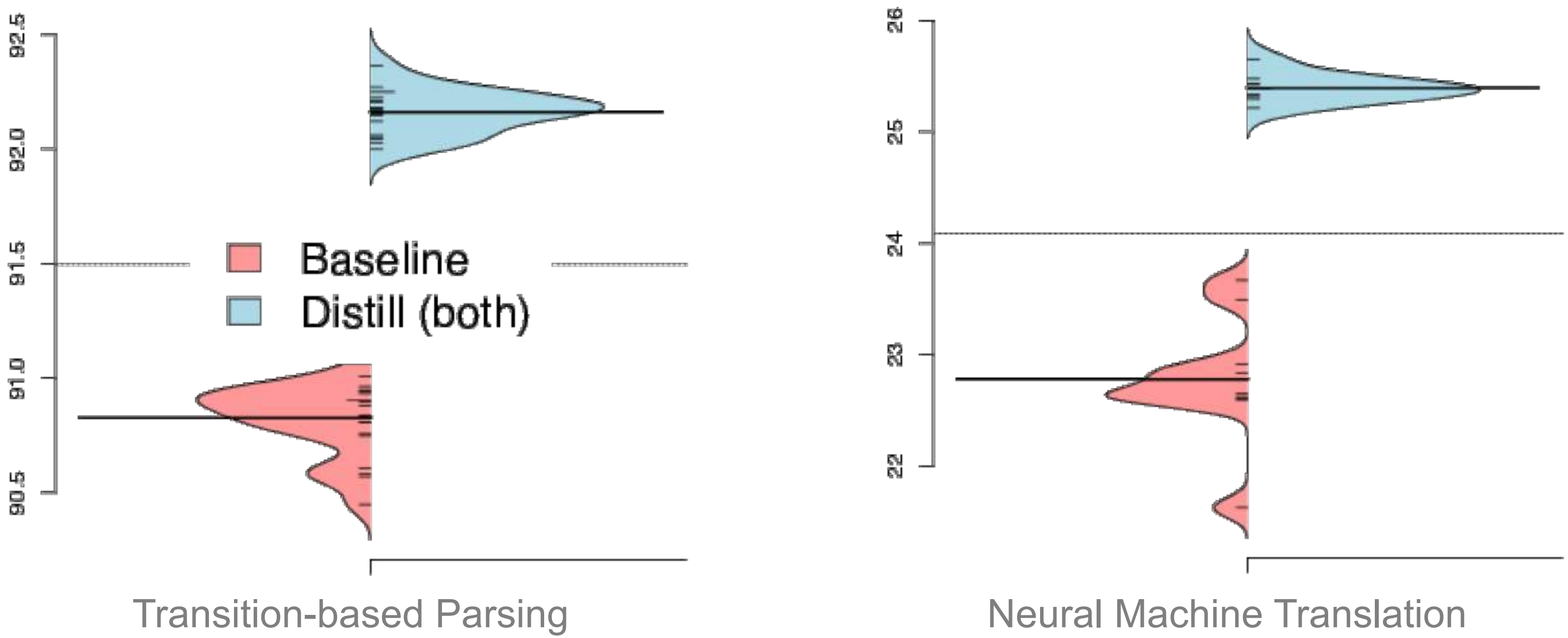
Results

Transition-based Dependency Parsing *Penn Treebank (Stanford dependencies)*	LAS
Baseline	90.83
Ensemble (20)	92.73
Distill (reference, $\alpha = 1.0$)	91.99
Distill (exploration)	92.00
Distill (both)	92.14
Ballesteros et al. (2016) (dyn. oracle)	91.42
Andor et al. (2016) (local, B=1)	91.02

Neural Machine Translation *IWSLT 2014 en-de*	BLEU
Baseline	22.79
Ensemble (10)	26.26
Distill (reference, $\alpha = 0.8$)	24.76
Distill (exploration)	24.64
Distill (both)	25.44
MIXER (Ranzato et al. 2015)	20.73
Wiseman and Rush (2016) (local B=1)	22.53
Wiseman and Rush (2016) (global B=1)	23.83

Yijia Liu, Wanxiang Che, Huaipeng Zhao, Bing Qin and Ting Liu. Distilling Knowledge for Search-based Structured Prediction. ACL 2018.

Analysis: Is Learning from KD Stable?



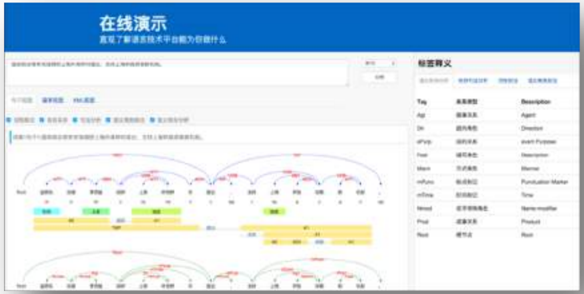
Yijia Liu, Wanxiang Che, Huaipeng Zhao, Bing Qin and Ting Liu. Distilling Knowledge for Search-based Structured Prediction. ACL 2018.

Part 3: Summary

- Transition-base Methods for Structured Prediction
- Neural Transition-base Methods
- Transition-base Methods with Beam-search Decoding
- Advanced Topics
  - Semantic dependency graph parsing
  - Multilingual dependency parsing
  - Knowledge Distillation

Part 4: Applications

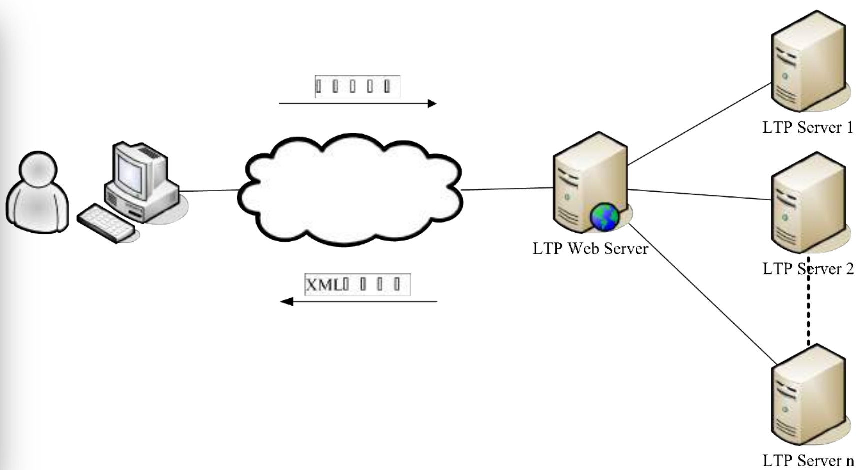
Language Technology Platform (LTP)



- http://ltp.ai
- Rich and accurate NLP toolkits
  - Chinese word segmentation,
  - POS tagging, NER, Dependency parsing,
  - Semantic role labeling, semantic dependency parsing
- Open source for research
- Evaluation
  - 1st place/13 at CoNLL 2009: syntactic and semantic dependency parsing
  - 1st place/27 at CoNLL 2018: multilingual syntactic dependency parsing

LTP-Cloud Service

- http://www.ltp-cloud.com/
- Advantages
  - Installation free, saving hardware, easy usage, cross-platform, cross-programming languages, update in time



Users of LTP-Cloud

- There are more than 10,000 users
- Response more than 700,000 requests each day

Awards

- 2016, the 1st prize of Heilongjiang Province Science and Technology Progress
- 2010, the 1st prize of Weichang Qian Chinese Information Processing Science and Technology Award

Our Consumers

- Tencent 腾讯
- Bai du 百度
- HUAWEI
- 中国移动 China Mobile
- 科大讯飞 iFLYTEK
- 搜狗搜索
- SONY make.believe
- SIEMENS
- GRIDSUM 国双 Empower your e-Performance
- 万方数据 WANFANG DATA 知识服务平台
- 同花顺 知识 · 工具 · 信息
- ANT'inno
- UBIC

How to Use Tree or Graph Structures?

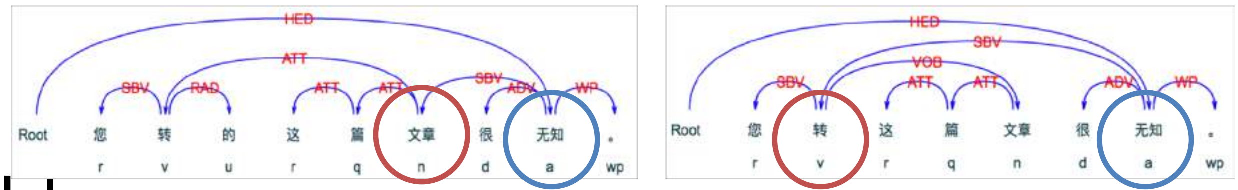
- As Information Extraction Rules
- As Input Features
- Multi-task Learning
- As Input Structures
- As Structured Prediction

How to Use Tree or Graph Structures?

- As Information Extraction Rules
- As Input Features
- Multi-task Learning
- As Input Structures
- As Structured Prediction

As Information Extraction Rules

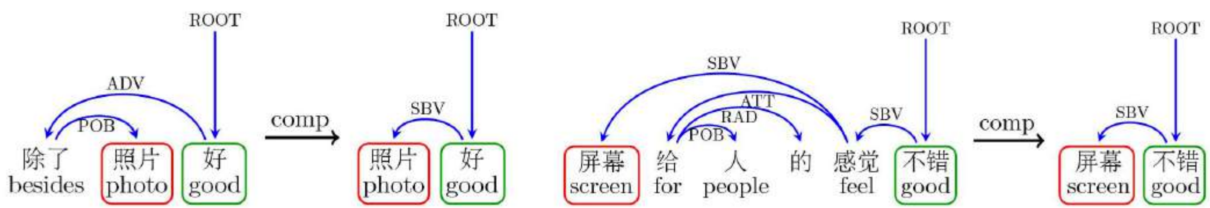
- For example
  - Polarity-target pair extraction



- Problem
  - The extraction rules are very complex
  - The parsing results are inexact

As Information Extraction Rules

- **Sentence compression** based PT pair extraction
  - Simplify the extraction rules
  - Improve the parsing accuracy



- Use a sequence labeling model to compress sentences
- The PT pair extraction performance improves 3%

Wanxiang Che, Yanyan Zhao, Honglei Guo, Zhong Su, Ting Liu. Sentence Compression for Aspect-Based Sentiment Analysis. IEEE/ACM Transactions on Audio, Speech, and Language Processing. 2015, 23(12)

How to Use Tree or Graph Structures?

- As Information Extraction Rules
- **As Input Features**
- Multi-task Learning
- As Input Structures
- As Structured Prediction

Path Features

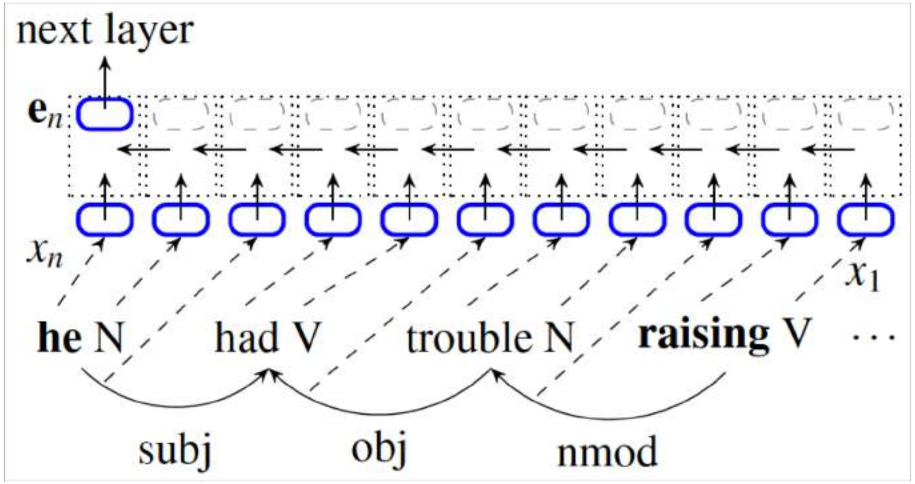
- For Example
  - Semantic Role Labeling (SRL), Relation Extraction (RC)

(a) Semantic Role Labeling. (b) Relation Classification.

- The parsing path features are very important
  - People <--> downtown: nsubj ← moved → nmod
- But they are difficult to be designed and very sparse

Path Features

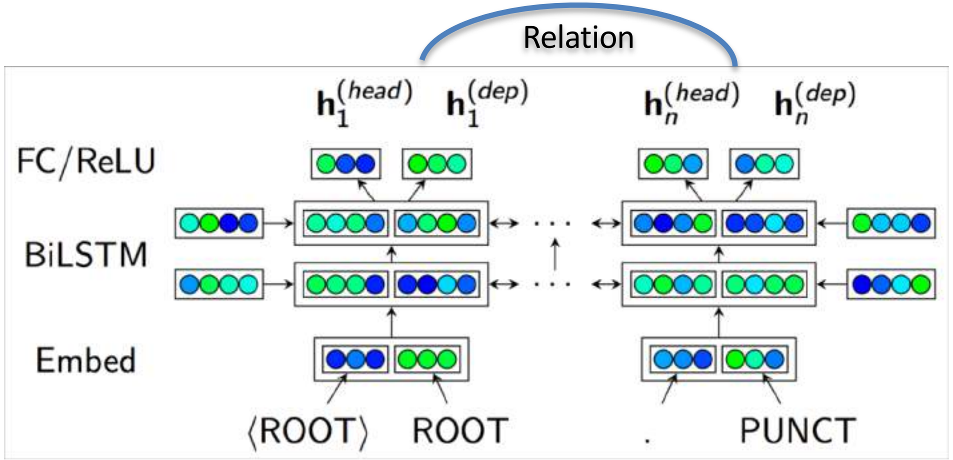
- Use LSTMs to represent paths
- All of word, POS tags and relations can be inputted



Michael Roth and Mirella Lapata. Neural Semantic Role Labeling with Dependency Path Embeddings. ACL 2016

Hidden Units of Parsing as Features

- The hidden units for parsing include **soft** syntactic information
- These can help applications, such as relation extraction



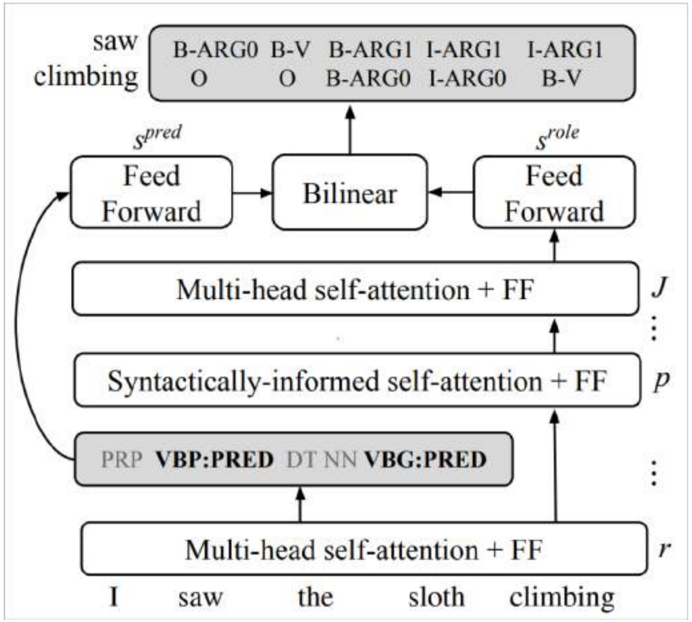
Meishan Zhang, Yue Zhang and Guohong Fu. End-to-End Neural Relation Extraction with Global Optimization. EMNLP 2017.

How to Use Tree or Graph Structures?

- As Information Extraction Rules
- As Input Features
- Multi-task Learning
- As Input Structures
- As Structured Prediction

Multi-task Learning

- MTL for Syntactic Parsing and Semantic Role Labeling



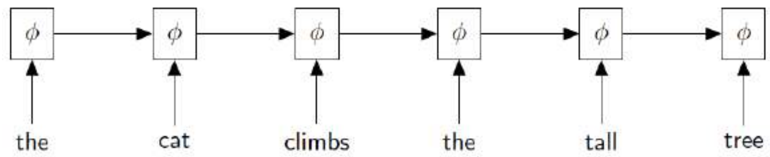
- Emma Strubell, Patrick Verga, Daniel Andor, David Weiss and Andrew McCallum. Linguistically-Informed Self-Attention for Semantic Role Labeling. EMNLP 2018.

How to Use Tree or Graph Structures?

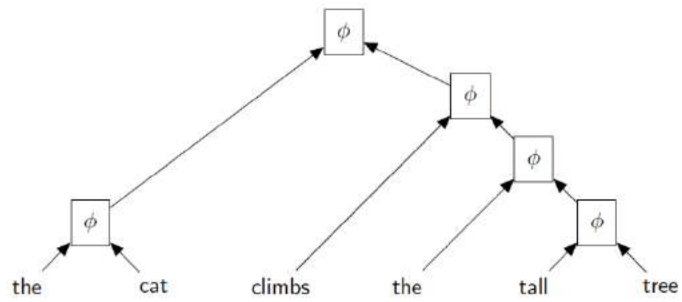
- As Information Extraction Rules
- As Input Features
- Multi-task Learning
- As Input Structures
- As Structured Prediction

Recurrent vs. Recursive Neural Networks

- Recurrent Neural Networks
  - Composing sequentially



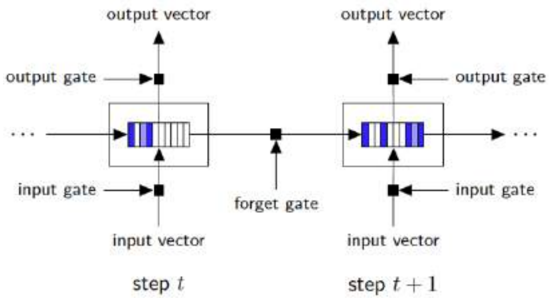
- Recursive Neural Networks
  - Use parse trees as input structures
  - Composing according to parsing structures



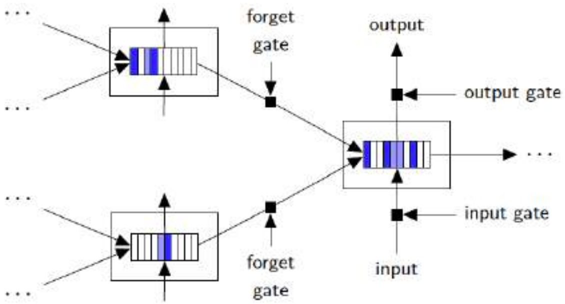
Richard Socher, Cliff Chiung-Yu Lin, Andrew Y. Ng and Christopher D. Manning. Parsing Natural Scenes And Natural Language With Recursive Neural Networks. ICML 2011.

Tree-LSTMs

- Standard LSTM

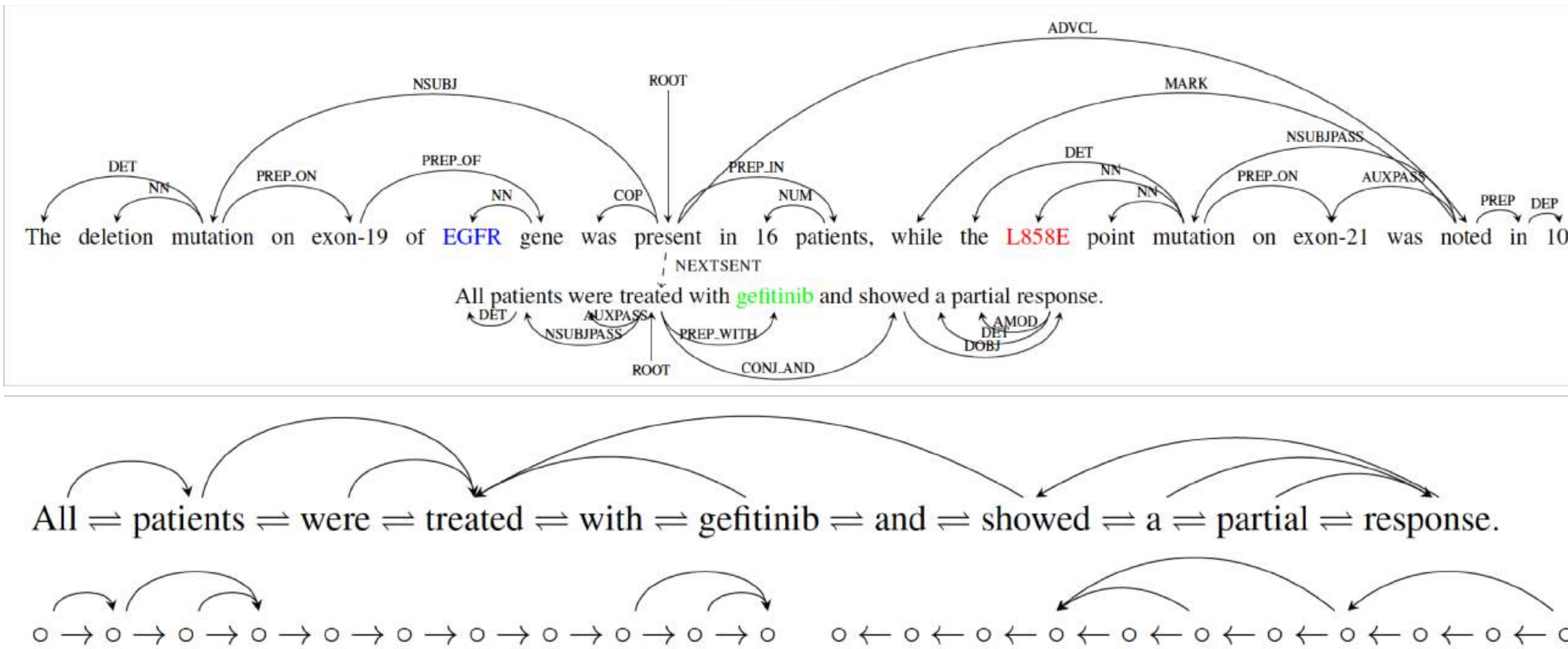


- Tree-LSTM



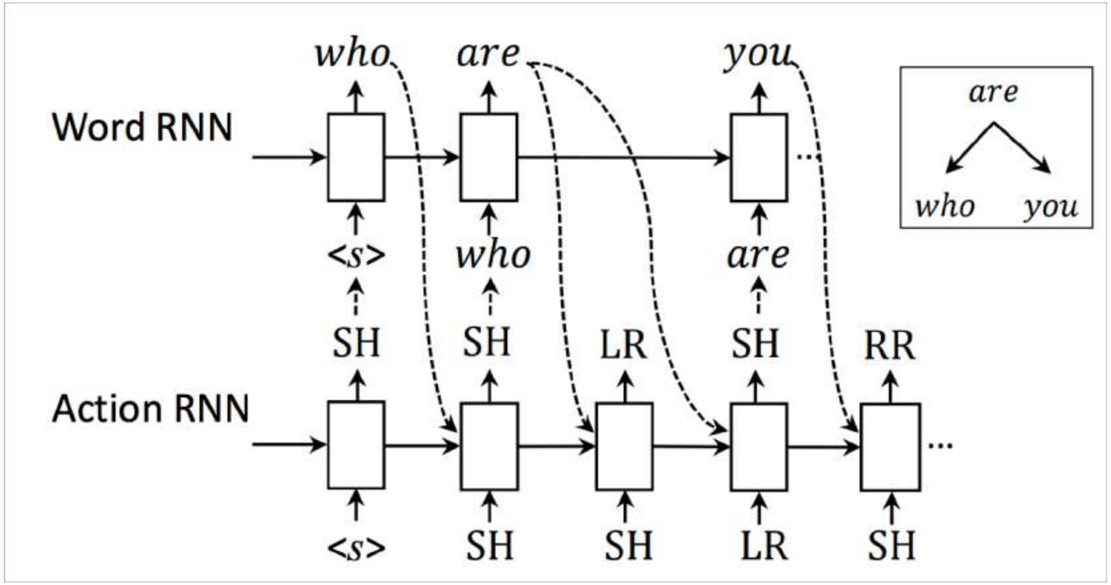
- Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. ACL 2015.
- Xiaodan Zhu, Parinaz Sobihani, and Hongyu Guo. 2015. Long short-term memory over recursive structures. ICML 2015.

Graph-LSTMs



Peng, N., Poon, H., Quirk, C., Toutanova, K., & Yih, W. 2017 Apr 5. Cross-Sentence N-ary Relation Extraction with **Graph LSTMs**. Transactions of the Association for Computational Linguistics.

Neural Machine Translation



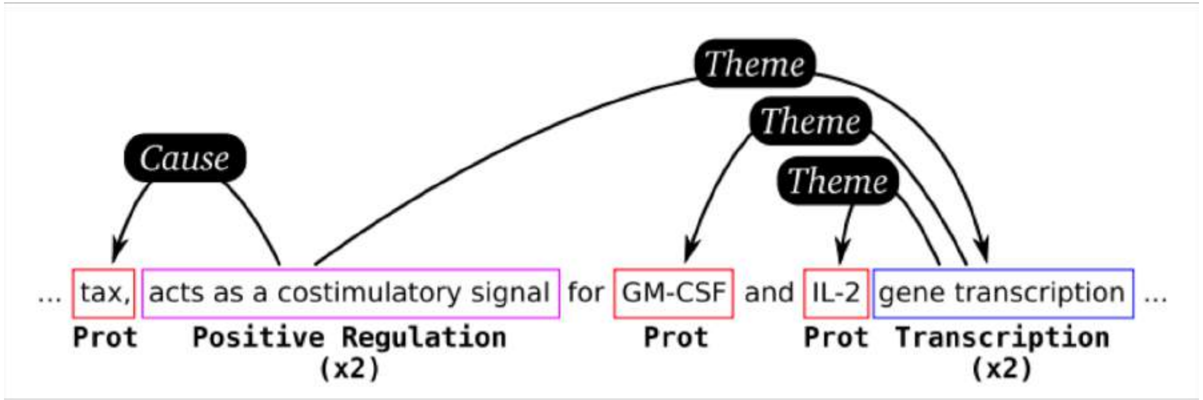
Dependency Decoder

Shuangzhi Wu, Dongdong Zhang, Nan Yang, Mu Li and Ming Zhou. Sequence-to-Dependency Neural Machine Translation. ACL 2017.

How to Use Tree or Graph Structures?

- As Information Extraction Rules
- As Input Features
- Multi-task Learning
- As Input Structures
- As Structured Prediction

Event Extraction



David McClosky, Mihai Surdeanu, and Christopher D. Manning. Event Extraction as Dependency Parsing. ACL 2011.

Disfluency Detection

- Disfluency detection for speech recognition

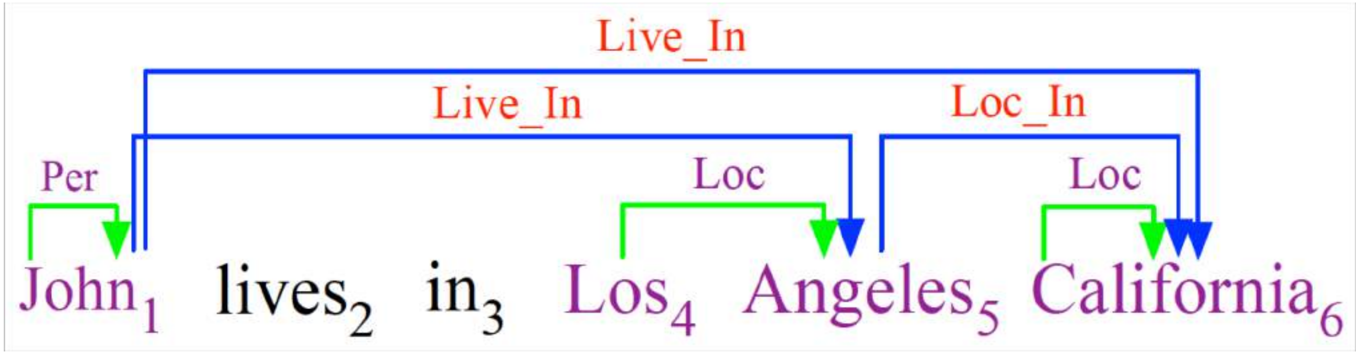
  I want a flight [$\underbrace{\textit{to Boston}}_{\text{RM}}$ + $\underbrace{\{\textit{um}\}}_{\text{IM}}$ $\underbrace{\text{to Denver}}_{\text{RP}}$]

- Transition System <*O*, *S*, *B*, *A*>
  - *output* (*O*) : represent the words that have been labeled as fluent
  - *stack* (*S*) : represent the partially constructed disfluency chunk
  - *buffer* (*B*) : represent the sentences that have not yet been processed
  - *action* (*A*) : represent the complete history of actions taken by the transition system
    - OUT: which moves the first word in the *buffer* to the output and clears out the *stack* if it is not empty
    - DEL: which moves the first word in the *buffer* to the *stack*

Shaolei Wang, Wanxiang Che, Yue Zhang, Meishan Zhang and Ting Liu. Transition-Based Disfluency Detection using LSTMs. EMNLP 2017

Entity Extraction and Classification

- Joint Entity Extraction and Classification
  - Convert the task into a directed graph

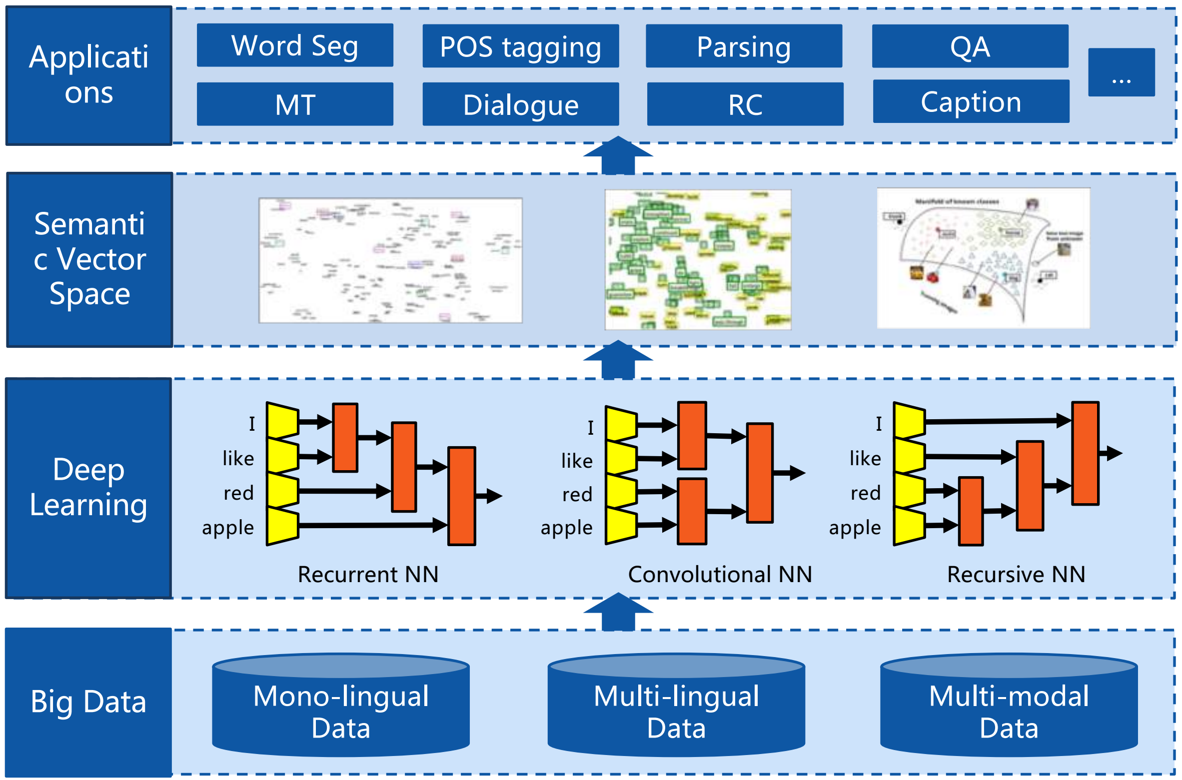


Shaolei Wang, Yue Zhang, Wanxiang Che and Ting Liu. Joint Extraction of Entities and Relations Based on a Novel Graph Scheme. IJCAI 2018.

Part 4: Summary

- LTP -- Language Technology Platform
- How to use tree or graph?
  - As Information Extraction Rules
  - As Input Features
  - Multi-task Learning
  - As Input Structures
  - As Structured Prediction

Deep Learning for NLP



Course Summarization

- Fundamental NLP Tasks
  - Lexical, Syntactic and Semantic Analysis
- Structured Prediction
  - Segmentation, Sequence Labeling and Parsing
- Methods
  - Graph-based and Transition-based
- Applications

Thanks!

- Welcome to SCIR!
  - http://ir.hit.edu.cn/

理解语言，认知社会
以中文技术，助民族复兴

长按二维码，关注哈工大SCIR
微信号：HIT_SCIR