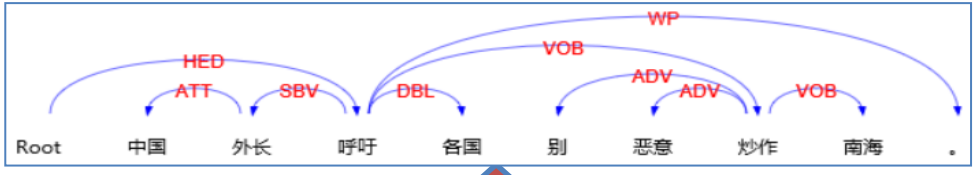
Part 6: Applications of Structure

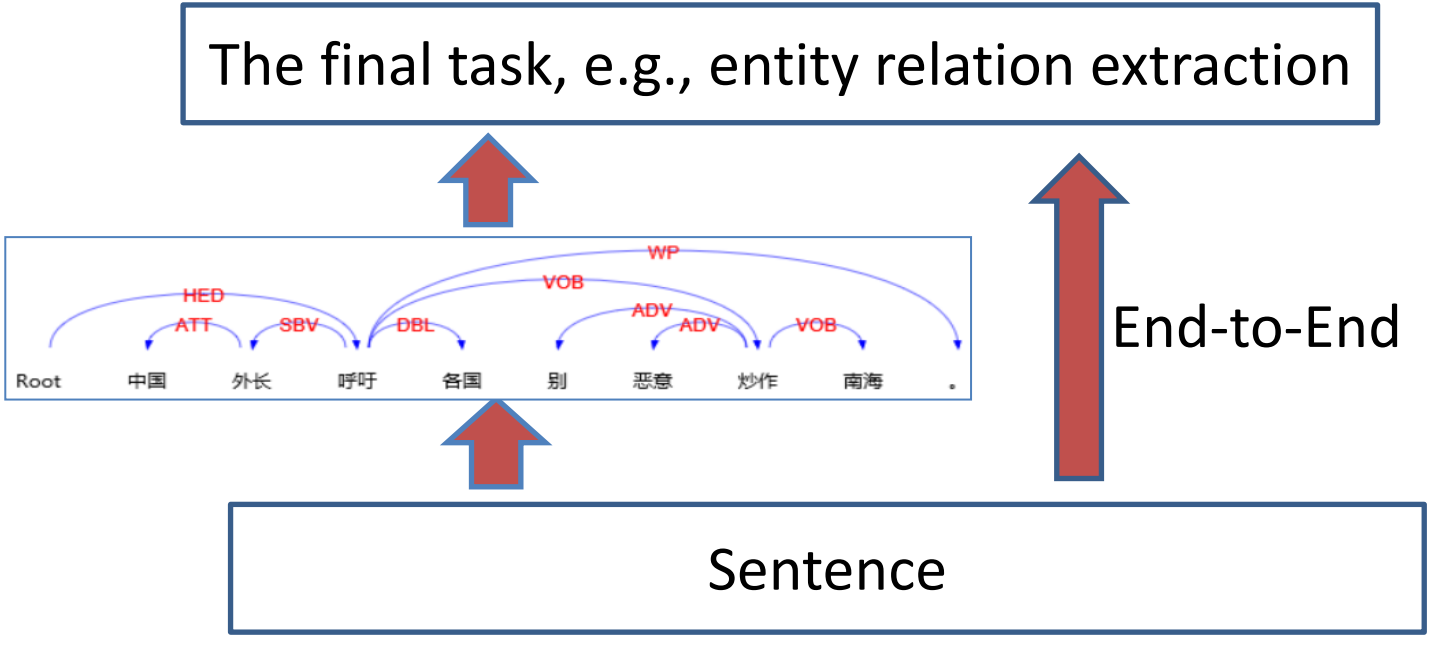
Shallow Learning

The final task, e.g., entity relation extraction



Sentence

Deep Learning



A Question

- Is Parsing or Structure Necessary?

	Bi-LSTM	Tree-LSTM
Stanford Sentiment TreeBank	49.8 / 50.7 (Segment)	50.4
Binary Sentiment Classification	79.0	77.4
Question-Answer Matching	56.4	55.8
Semantic Relationship Classification	**75.2**	**76.7**
Discourse Parsing	57.5	56.4

Jiwei Li, Minh-Thang Luong, Dan Jurafsky and Eduard Hovy. When Are Tree Structures Necessary for Deep Learning of Representations? EMNLP, 2015.

How to Use Tree or Graph Structures?

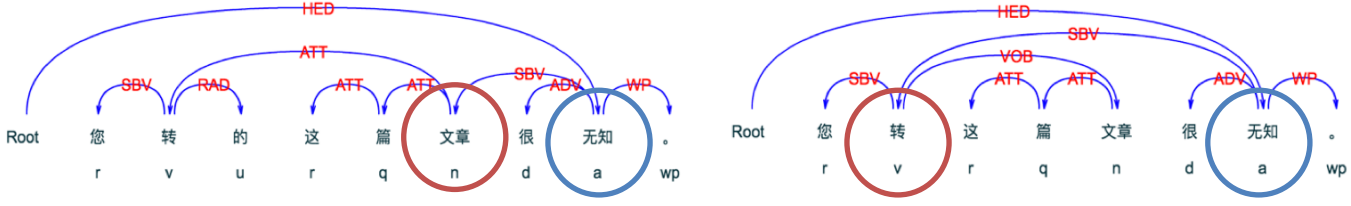
- As Information Extraction Rules
- As Input Features
- As Input Structures
- As Structured Prediction

How to Use Tree or Graph Structures?

- As Information Extraction Rules
- As Input Features
- As Input Structures
- As Structured Prediction

As Information Extraction Rules

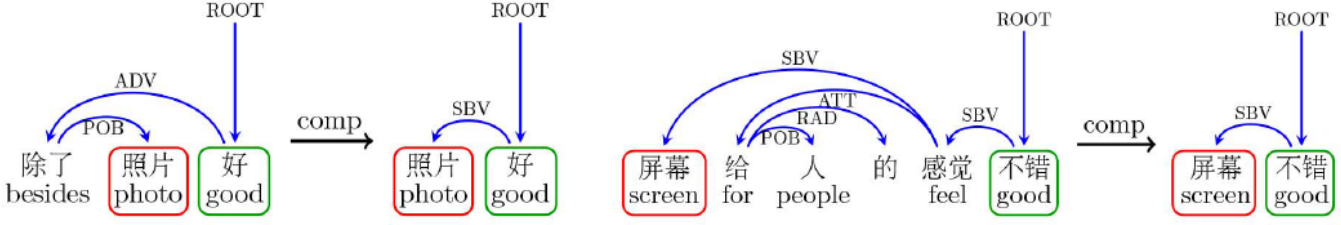
- For example
  - Polarity-target pair extraction



- 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%

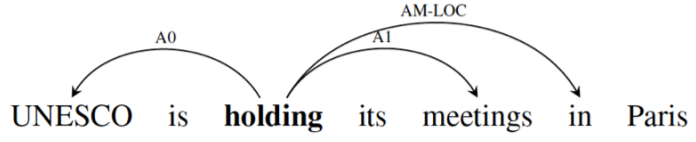
Wanxiang Che, Yanyan Zhao, Honglei Guo, Zhong Su, Ting Liu. Sentence Compression for Aspect-Based Sentiment Analysis. IEEE/ACM Transactions on Audio, Speech, and Language Processing. 2015, 23(12)

How to Use Tree or Graph Structures?

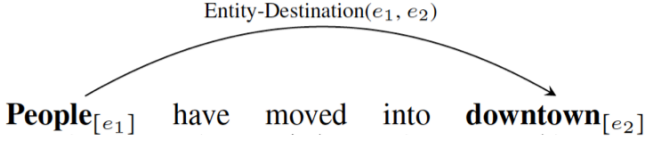
- As Information Extraction Rules
- As Input Features
- As Input Structures
- As Structured Prediction

Path Features

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



(a) Semantic Role Labeling.

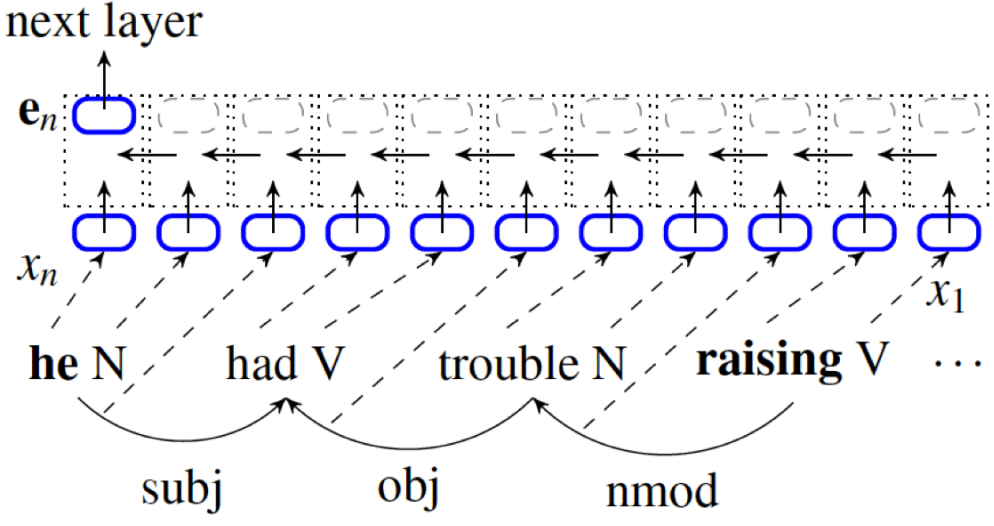


(b) Relation Classification.

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

Path Features

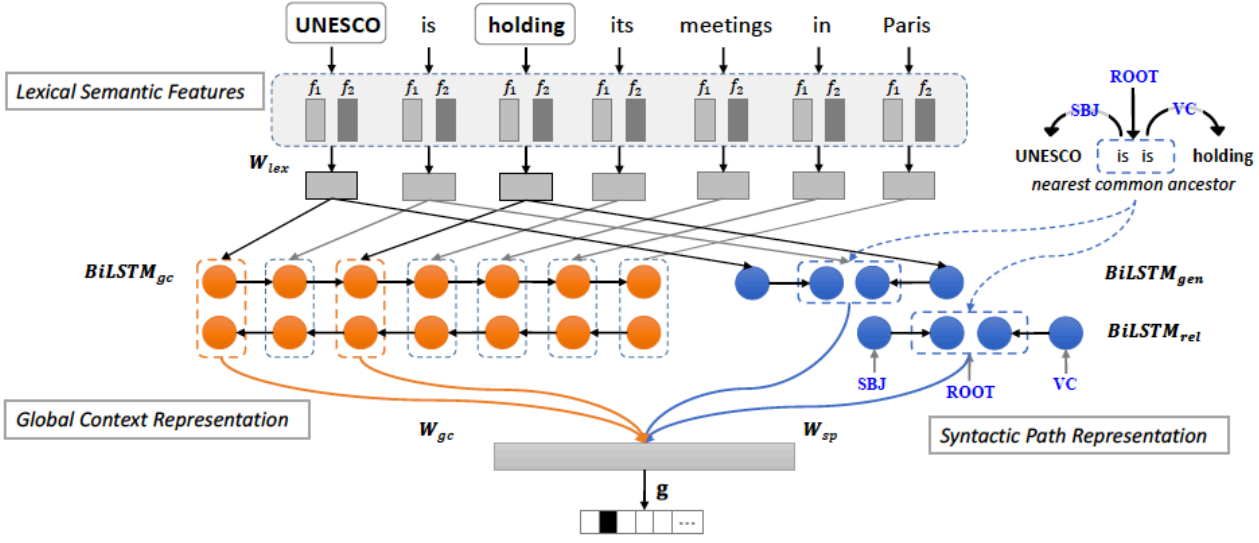
- Use LSTMs to represent paths
- All of word, POS tags and relations can be inputted



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

Joint learning of SRL and RC

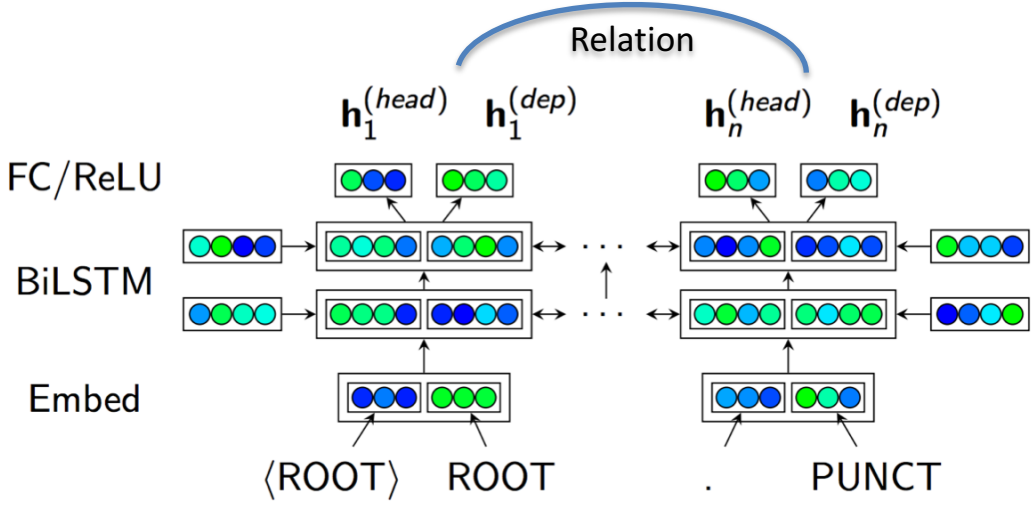
- Multi-task learning



Jiang Guo, Wanxiang Che, Haifeng Wang and Ting Liu. A Unified Architecture for Semantic Role Labeling and Relation Classification. Coling 2016.

Hidden Units of Parsing as Features

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



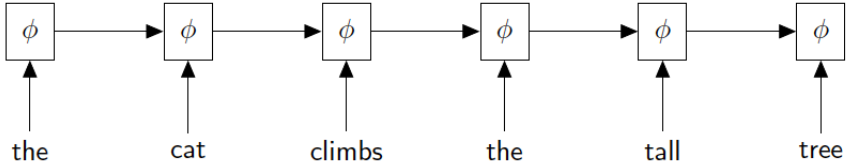
Meishan Zhang, Yue Zhang and Guohong Fu. End-to-End Neural Relation Extraction with Global Optimization. EMNLP 2017.

How to Use Tree or Graph Structures?

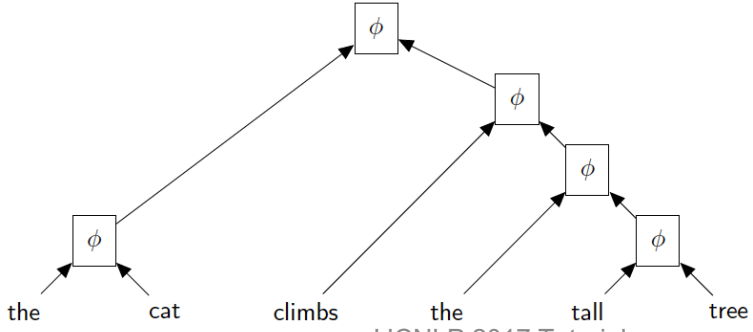
- As Information Extraction Rules
- As Input Features
- As Input Structures
- As Structured Prediction

Recurrent vs. Recursive Neural Networks

- Recurrent Neural Networks
  - Composing sequentially



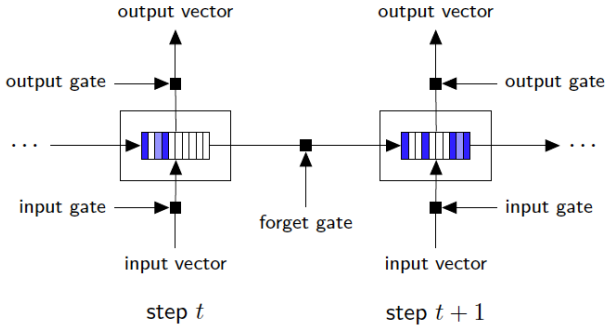
- Recursive Neural Networks
  - Use parse trees as input structures
  - Composing according to parsing structures



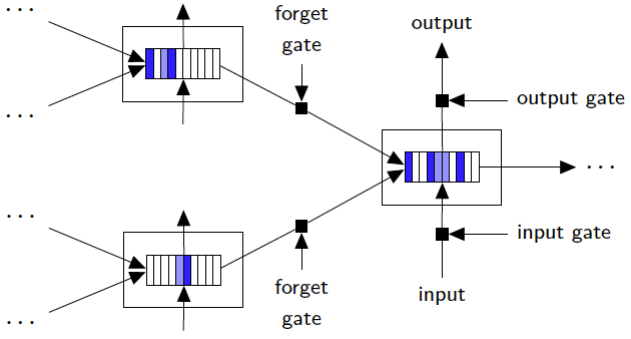
Richard Socher, Cliff Chiung-Yu Lin, Andrew Y. Ng and Christopher D. Manning. Parsing Natural Scenes And Natural Language With Recursive Neural Networks. ICML 2011.

Tree-LSTMs

- Standard LSTM

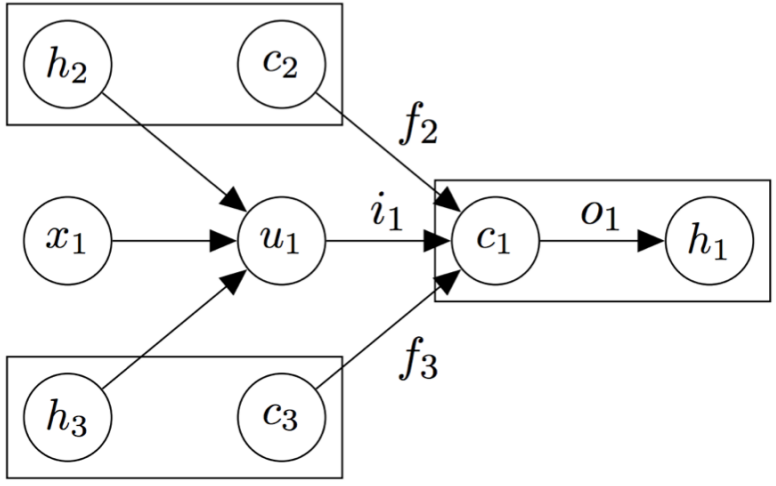


- Tree-LSTM



- 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.

Tree-LSTMs



$$\tilde{h}_j = \sum_{k \in C(j)} h_k,$$

$$i_j = \sigma\left(W^{(i)} x_j + U^{(i)} \tilde{h}_j + b^{(i)}\right),$$

$$f_{jk} = \sigma\left(W^{(f)} x_j + U^{(f)} h_k + b^{(f)}\right),$$

$$o_j = \sigma\left(W^{(o)} x_j + U^{(o)} \tilde{h}_j + b^{(o)}\right),$$

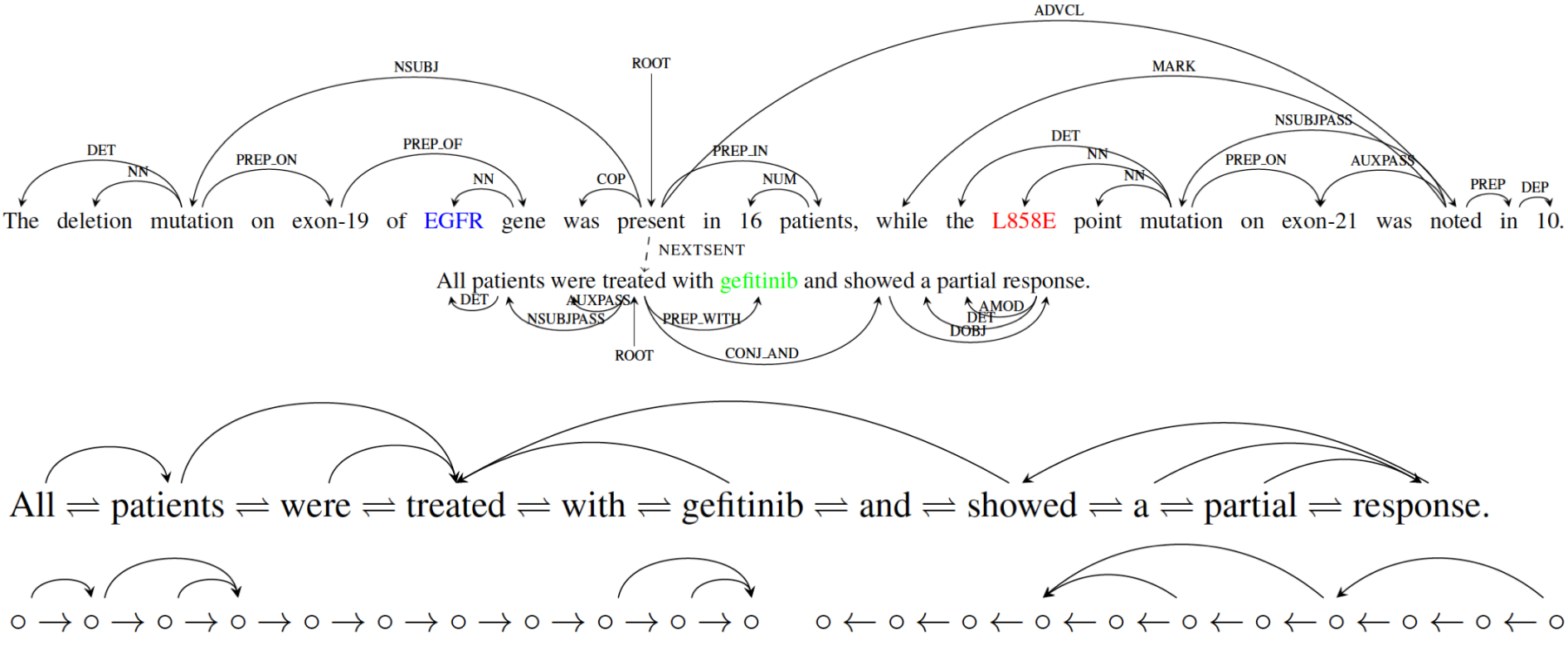
$$u_j = \tanh\left(W^{(u)} x_j + U^{(u)} \tilde{h}_j + b^{(u)}\right),$$

$$c_j = i_j \odot u_j + \sum_{k \in C(j)} f_{jk} \odot c_k,$$

$$h_j = o_j \odot \tanh(c_j),$$

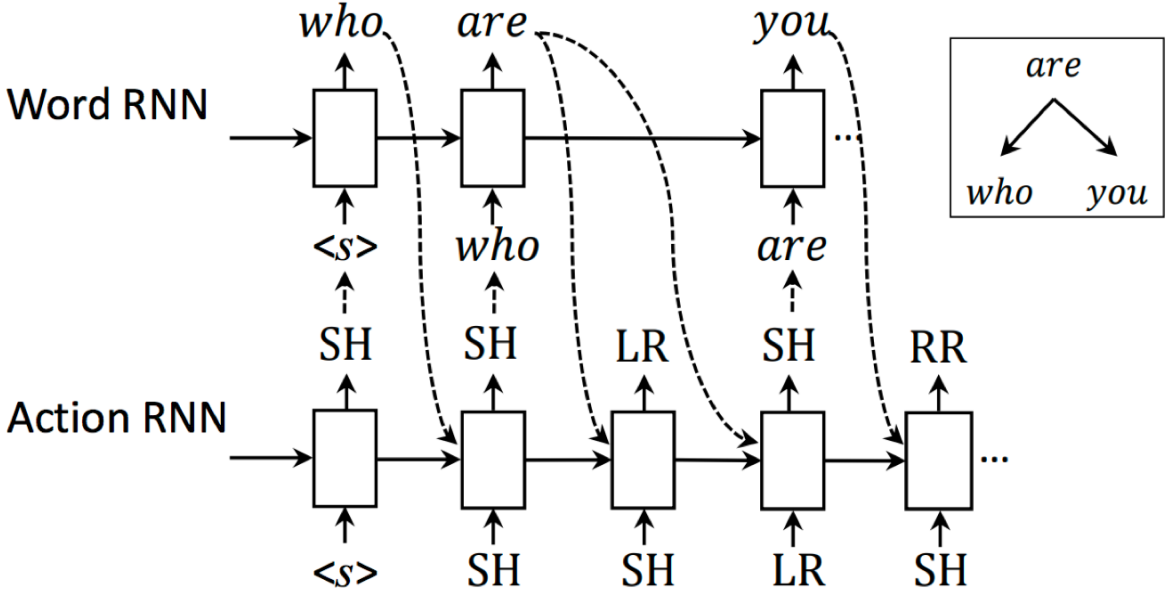
- Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. ACL 2015.
- Xiaodan Zhu, Parinaz Sobihani, and Hongyu Guo. 2015. Long short-term memory over recursive structures. ICML 2015.

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

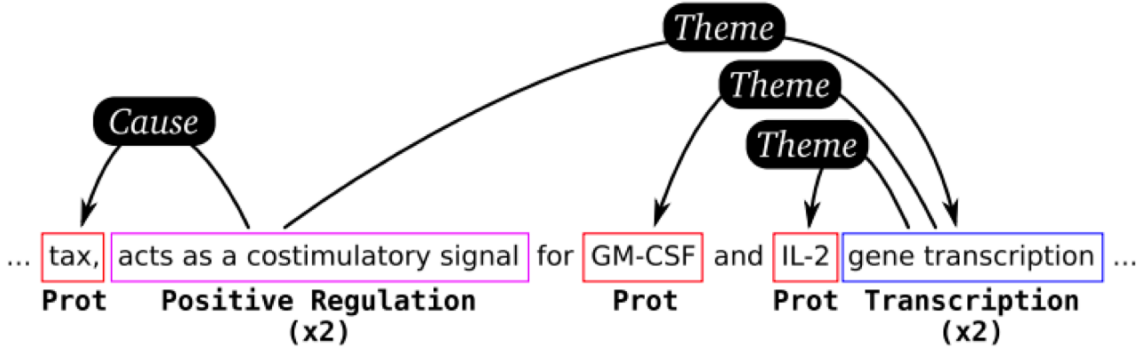
Shuangzhi Wu, Dongdong Zhang, Nan Yang, Mu Li and Ming Zhou. Sequence-to-Dependency Neural Machine Translation. ACL 2017.

How to Use Tree or Graph Structures?

- As Information Extraction Rules
- As Input Features
- As Input Structures
- As Structured Prediction

Event Extraction

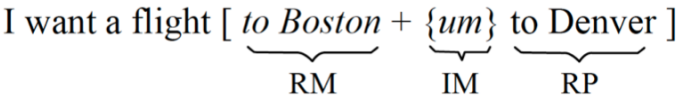
- Event Extraction as Dependency Parsing



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

Disfluency Detection

- Disfluency detection for speech recognition



- 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.

Disfluency Detection

- An Example of transition-based disfluency detection

Step	Action	Output	Stack	Buffer
0		[]	[]	[a, flight, to, boston, to, denver]
1	OUT	[a]	[]	[flight, to, boston, to, denver]
2	OUT	[a, flight]	[]	[to, boston, to, denver]
3	DEL	[a, flight]	[to]	[boston, to, denver]
4	DEL	[a, flight]	[to, boston]	[to, denver]
5	OUT	[a, flight, to]	[]	[denver]
6	OUT	[a, flight, to, denver]	[]	[]

- Results

Method	**P**	**R**	**F1**
Our	91.1	84.1	**87.5**
Attention-based (Wang et al., 2016)	91.6	82.3	86.7
Bi-LSTM (Zayats et al., 2016)	91.8	80.6	85.9
semi-CRF (Ferguson et al., 2015)	90.0	81.2	85.4
UBT (Wu et al., 2015)	90.3	80.5	85.1
M^3N (Qian and Liu, 2013)	-	-	84.1

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

Summary

- As Information Extraction Rules
- As Input Features
- As Input Structures
- As Structured Prediction

Course Summarization

- Lexical, Syntactic and Semantic Analysis
  - Structured Prediction (Segmentation, Tagging and Parsing)
- Deep Learning
  - Representation Learning
  - End-to-end Learning
- Traditional Methods
  - Graph-based and Transition-based
- Neural Network Methods
  - Graph-based and Transition-based
- Applications