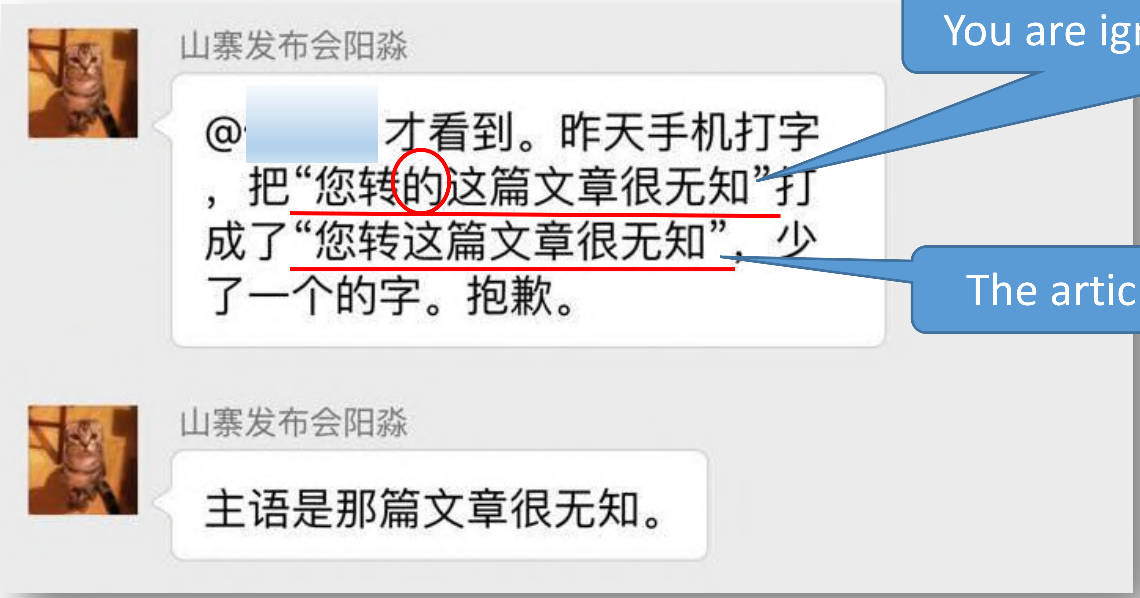
Syntactic and Semantic Parsing

Wanxiang Che
Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology

2017-12-1

Why Do We Need Parsing?



You are ignorant to retweet the article

The article you retweeted is ignorant

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

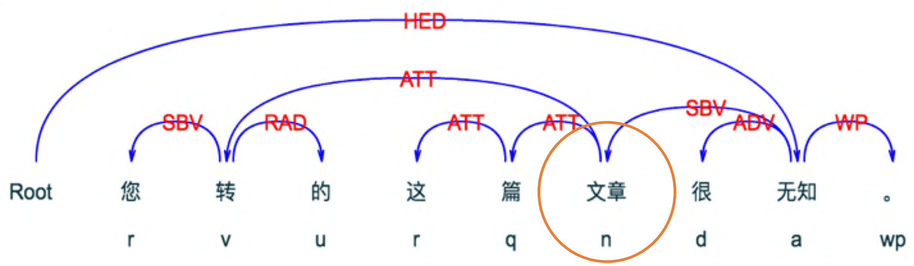
Outline

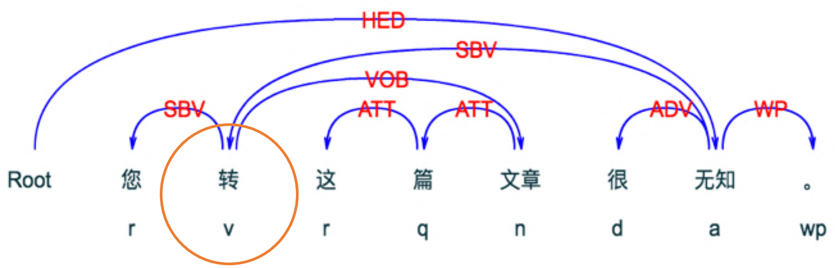
1. Syntactic and Semantic Parsing
2. Pseudo Data for Parsing
3. Applications of Parsing
4. Summary

Outline

Syntactic and Semantic Parsing

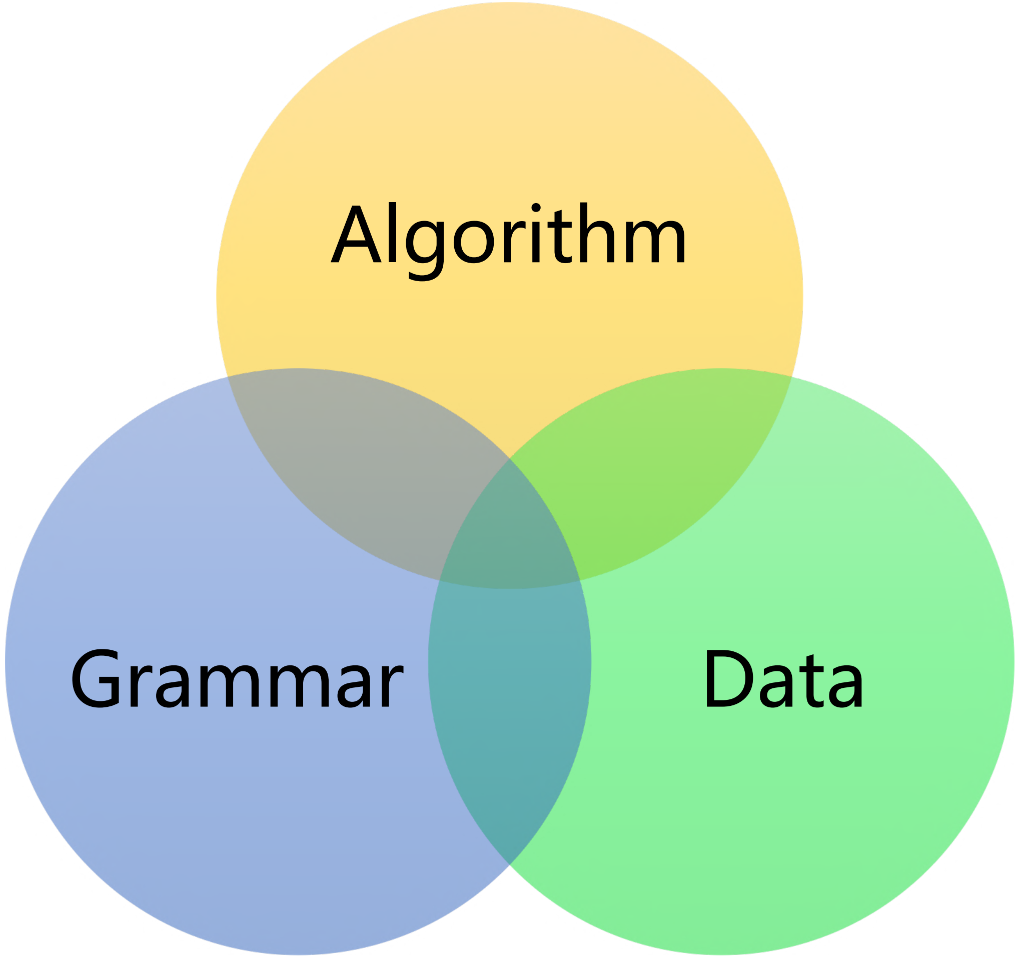
- The analysis of a sentence into its constituents, resulting in a parse tree or graph showing their syntactic or semantic relation to each other
- A traditional and core NLP task



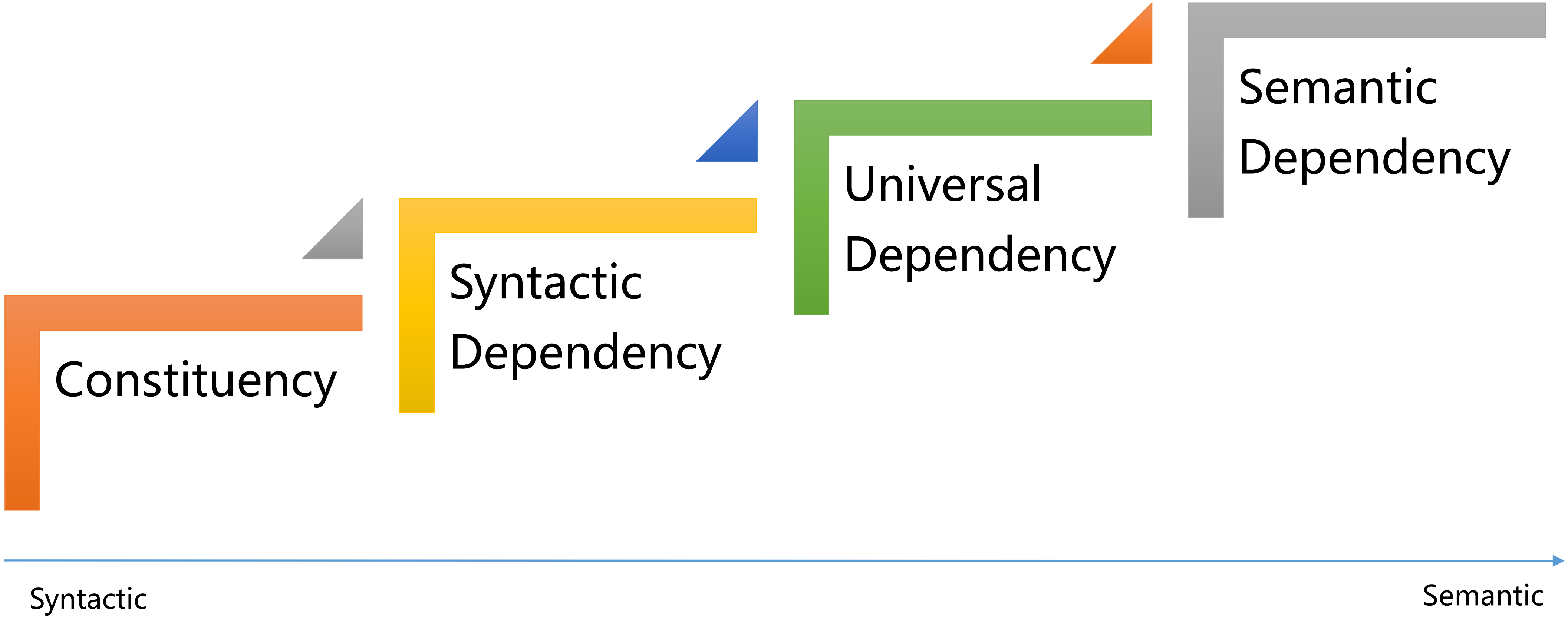


http://ltp.ai/demo.html

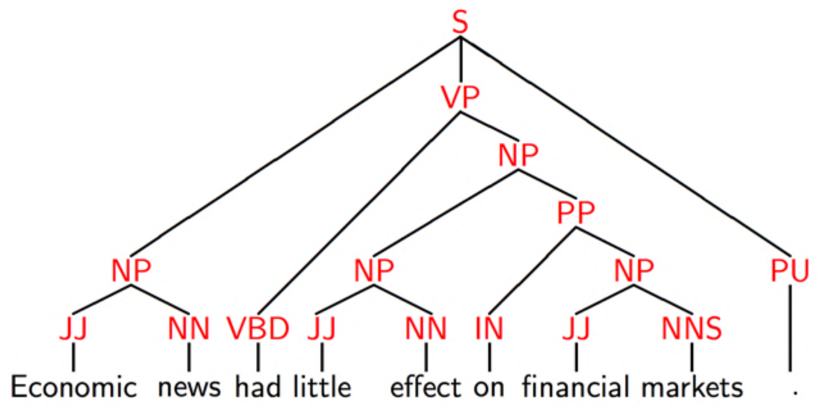
Components of Parsing



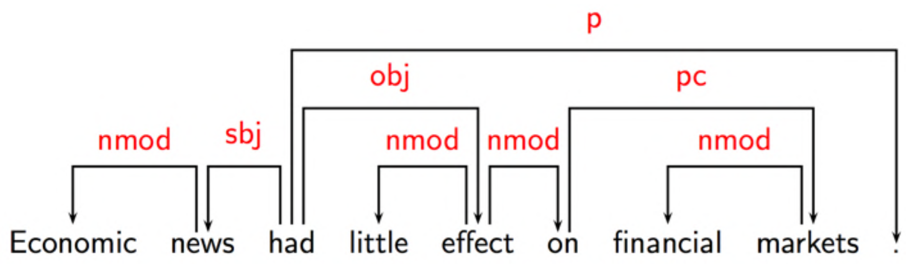
Grammar


Constituency
Syntactic Dependency
Universal Dependency
Semantic Dependency
Syntactic
Semantic

Constituency vs. Dependency



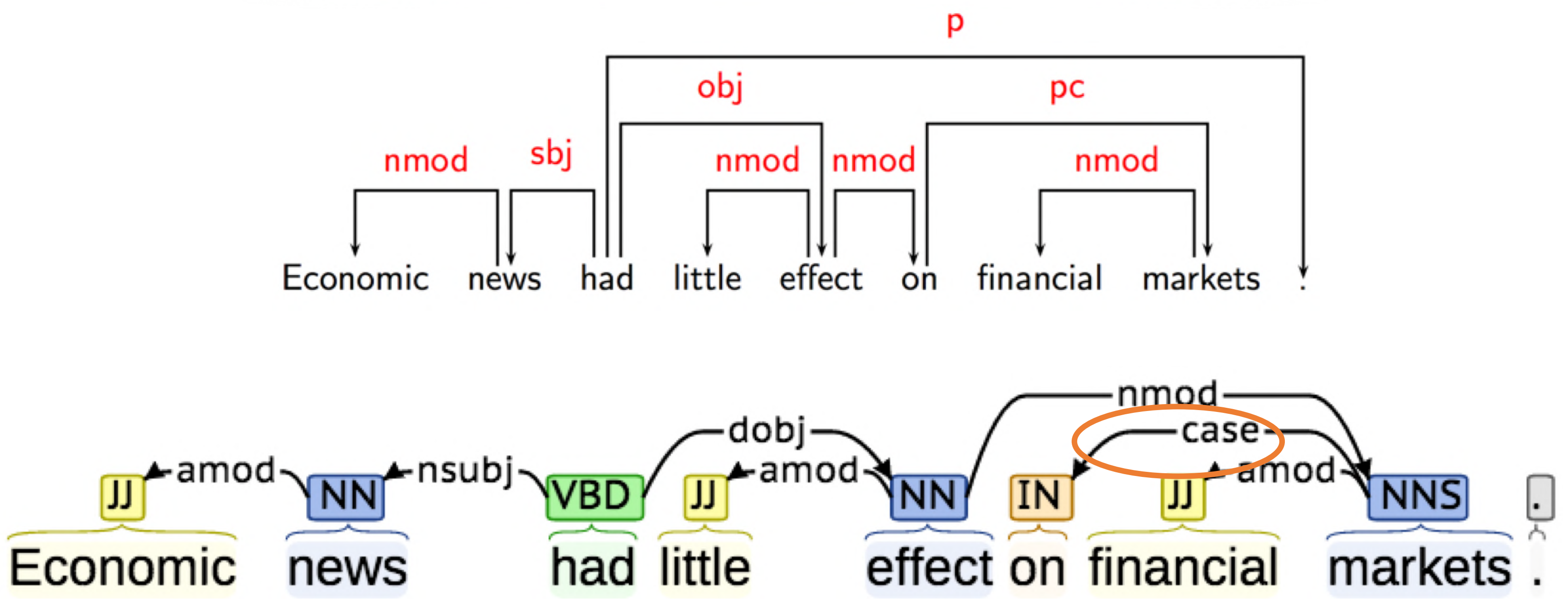
Constituency



Dependency

- Dependency Structures
  - Usually easier to be understood
  - More amenable to annotators

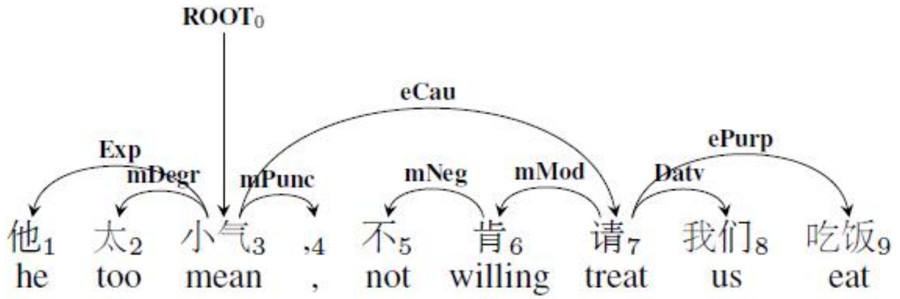
Syntactic vs. Universal Dependency



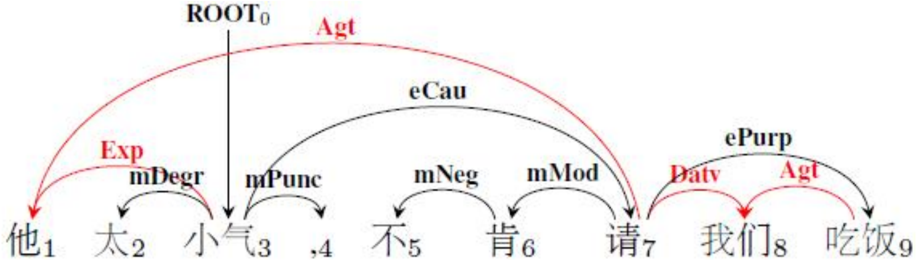
http://nlp.stanford.edu/software/stanford-dependencies.shtml

- Universal Dependencies pay more attention to relations between content words
- The universal annotation scheme for all languages

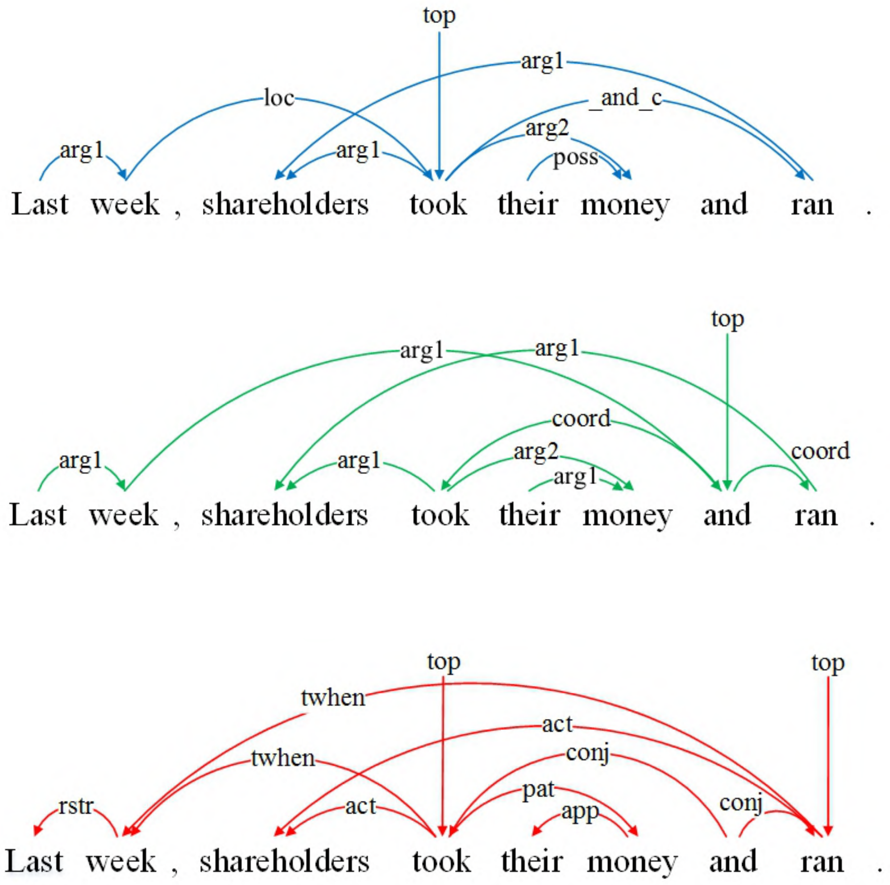
Semantic Dependency Tree and Graph



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)

Data

Monolingual Single-domain

Multilingual

Multi-domain

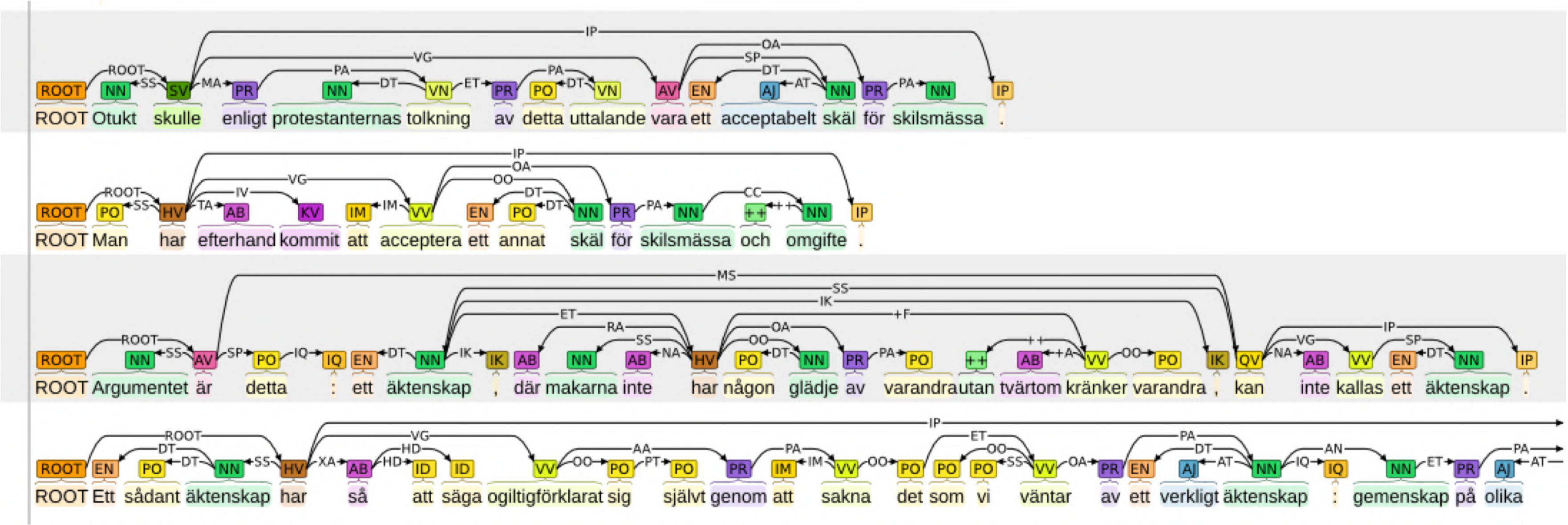
Universal Treebank

Semantic Dependency Treebank

Rich-resource → Low-resource

Multilingual Treebanks

- CoNLL 2006, 2007 Shared Tasks
  - http://ilk.uvt.nl/conll/
  - 10 - 12 Languages



Syntactic and Semantic Dependencies in Multiple Languages

- CoNLL 2009 Shared Task
  - http://ufal.mff.cuni.cz/conll2009-st/
  - Syntactic and Semantic Dependencies in Multiple Languages
  - 7 Languages
  - We achieved Rank 1

Rank	System	Average	Catalan	Chinese	Czech	English	German	Japanese	Spanish
1	Che	82.64	81.84	@ 76.38	@ 83.27	87.00	@ 82.44	@ 85.65	81.90
2	Chen	82.52	@ 83.01	76.23	80.87	@ 87.69	81.22	85.28	@ 83.31
3	Merlo	82.14	82.66	76.15	83.21	86.03	79.59	84.91	82.43
4	Bohnet	80.85	80.44	75.91	79.57	85.14	81.60	82.51	80.75
5	Asahara	78.43	75.91	73.43	81.43	86.40	69.84	84.86	77.12
6	Brown	77.27	77.40	72.12	75.66	83.98	77.86	76.65	77.21
7	Zhang	76.49	75.00	73.42	76.93	82.88	73.76	78.17	75.25
8	Dai	73.98	72.09	72.72	67.14	81.89	75.00	80.89	68.14
9	Lu Li	73.97	71.32	65.53	75.85	81.92	70.93	80.49	71.72
10	Lluís	71.49	56.64	66.18	75.95	81.69	72.31	81.76	65.91
11	Vallejo	70.81	73.75	67.16	60.50	78.19	67.51	77.75	70.78
12	Ren	67.81	59.42	75.90	60.18	77.83	65.77	77.63	57.96
13	Zeman	51.07	49.61	43.50	57.95	50.27	49.57	57.69	48.90

Multiple Domain

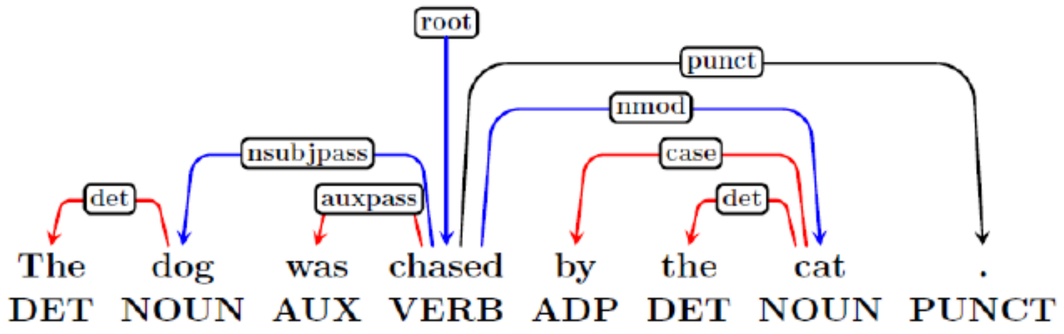
- Syntactic Analysis of Non-Canonical Language (SANCL) 2012 Shared Task
  - https://sites.google.com/site/sancl2012/
  - Organized by Google
  - Data: Google Web Treebank (CQA, Newsgroup, Online Review)
  - We achieved Rank 2 (Stanford) and 3 (HIT)

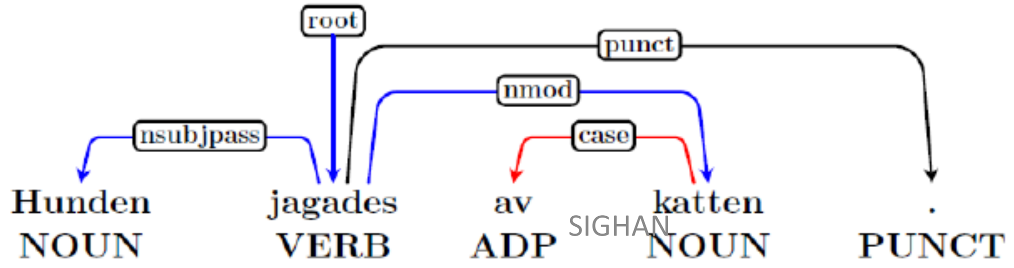
Dependency Parsing Results:

	Domain A (answers)			Domain B (newsgroups)			Domain C (reviews)			Domain D (wsj)			Average (A-C)		
Team	LAS	UAS	POS	LAS	UAS	POS	LAS	UAS	POS	LAS	UAS	POS	LAS	UAS	POS
*Zhang&Nivre**	*76.60*	*81.59*	*89.74*	*81.62*	*85.19*	*91.17*	*78.10*	*83.32*	*89.60*	*89.37*	*91.46*	*96.84*	*78.77*	*83.37*	*90.17*
UPenn	68.54	82.28	89.65	74.41	86.10	90.99	70.17	82.88	89.02	81.74	91.99	96.93	71.04	83.75	89.89
UMass	72.51	78.36	89.42	77.23	81.61	91.28	74.89	80.34	89.90	81.15	83.97	94.71	74.88	80.10	90.20
NAIST	73.54	79.89	89.92	79.83	84.59	91.39	75.72	81.99	90.47	87.95	90.99	97.40	76.36	82.16	90.59
IMS-2	74.43	80.77	89.50	79.63	84.29	90.72	76.55	82.18	89.41	86.88	89.90	97.02	76.87	82.41	89.88
IMS-3	75.90	81.30	88.24	79.77	83.96	89.70	77.61	82.38	88.15	86.02	88.89	95.14	77.76	82.55	88.70
IMS-1	78.33	83.20	91.07	83.16	86.86	91.70	79.02	83.82	90.01	90.82	92.73	97.57	80.17	84.63	90.93
Copenhagen	78.12	82.91	90.42	82.90	86.59	91.15	79.58	84.13	89.83	90.47	92.42	97.25	80.20	84.54	90.47
Stanford-2	77.50	82.57	90.30	83.56	87.18	91.49	79.70	84.37	90.46	89.87	91.95	95.00	80.25	84.71	90.75
HIT-Baseline	80.75	85.84	90.99	85.26	88.90	92.32	81.60	86.60	90.65	**91.88**	**93.88**	**97.76**	82.54	87.11	91.32
HIT-Domain	80.79	85.86	90.99	85.18	88.81	92.32	81.92	86.80	90.65	91.82	93.83	97.76	82.63	87.16	91.32
Stanford-1	81.01	85.70	90.30	**85.85**	**89.10**	91.49	82.54	86.73	90.46	91.50	93.38	95.00	83.13	87.18	90.75
DCU-Paris13	**81.15**	**85.80**	**91.79**	85.38	88.74	**93.81**	**83.86**	**88.31**	**93.11**	89.67	91.79	97.29	**83.46**	**87.62**	**92.90**

Universal Treebank

- Universal Dependencies and POS Tags
  - http://universaldependencies.org/
  - 50+ Languages, 70+ Treebanks





CoNLL 2017 Shared Task

- http://universaldependencies.org/conll17/
- Multilingual Parsing from Raw Text to Universal Dependencies
  - Tasks: Sentence Segmentation, Word Segmentation, POS Tagging, Parsing
  - Training: 45 languages, 64 treebanks
  - Test: 81 treebanks
- 113 Registration Teams
  - Universities: Stanford, CMU, UW, Cornell, Toronto, Cambridge, Tokyo, …
  - Companies: IBM Research, Facebook, …
  - China: CAS, Fudan, Shanghai Jiaotong, …
- Results
  - 33 Submission Teams
  - Rank 1-3: Stanford, Cornell, Stuttgart
  - HIT Rank 4

Semantic Dependency Parsing

- We organized SemEval 2012 and 2016 Shared Tasks
  - https://www.cs.york.ac.uk/semeval-2012/task5.html
  - http://alt.qcri.org/semeval2016/task9/
  - Chinese Semantic Dependency Parsing
- SemEval 2014 and 2015
  - http://alt.qcri.org/semeval2014/task8/
  - http://alt.qcri.org/semeval2015/task18/
  - English Semantic Dependency Parsing

Algorithm

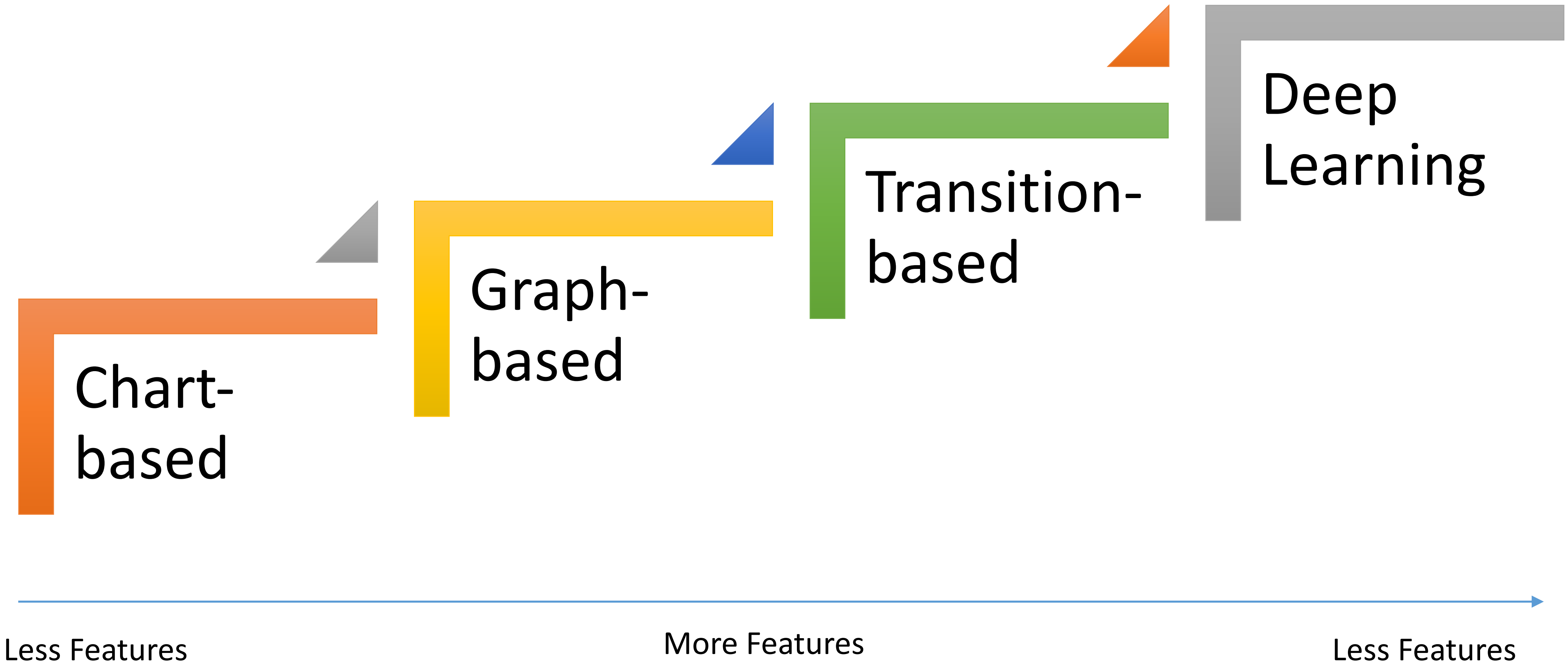
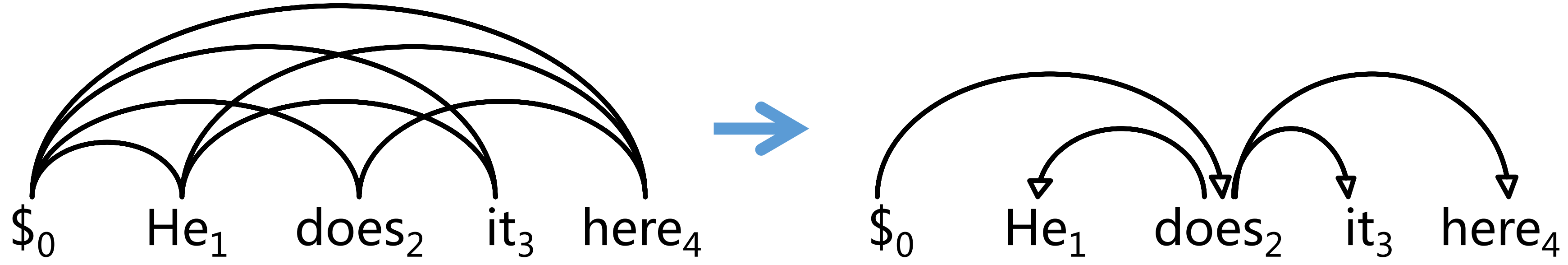


Chart-
based
Graph-
based
Transition-
based
Deep
Learning
Less Features
More Features
Less Features

Graph-based Dependency Parsing

- Find the highest scoring tree from a complete dependency graph
- Maximum Spanning Tree (MST)
  - Some dynamic programming algorithms



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

How to Calculate the Score of a Tree

- The score of a tree is the sum of each arc

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

- An arc is represented as a feature vector

$$score(h, m) = ?$$

$\$_0$ He$_1$ does$_2$ it$_3$ here$_4$

- The score of the arc is dot product of weight vector by feature vector

$$score(h, m) = \mathbf{w}\cdot\mathbf{f}(h, m)$$

Features for an Arc

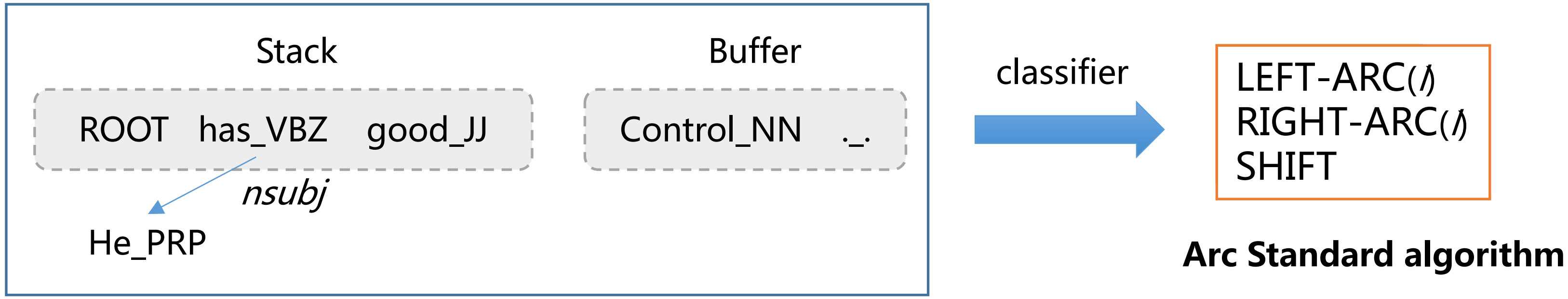
* As McGwire neared , fans went wild

[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]

Transition-based Dependency Parsing

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

  = Stack + Buffer + Dependency Arcs



Configuration

Traditional Features

Configuration

Need Tedious Feature Engineering!

Feature
- Binary
- Sparse
- High-dimensional

0	0	1	0	1	...	0	1	0	0

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_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.

Neural Network Parser

Softmax layer:

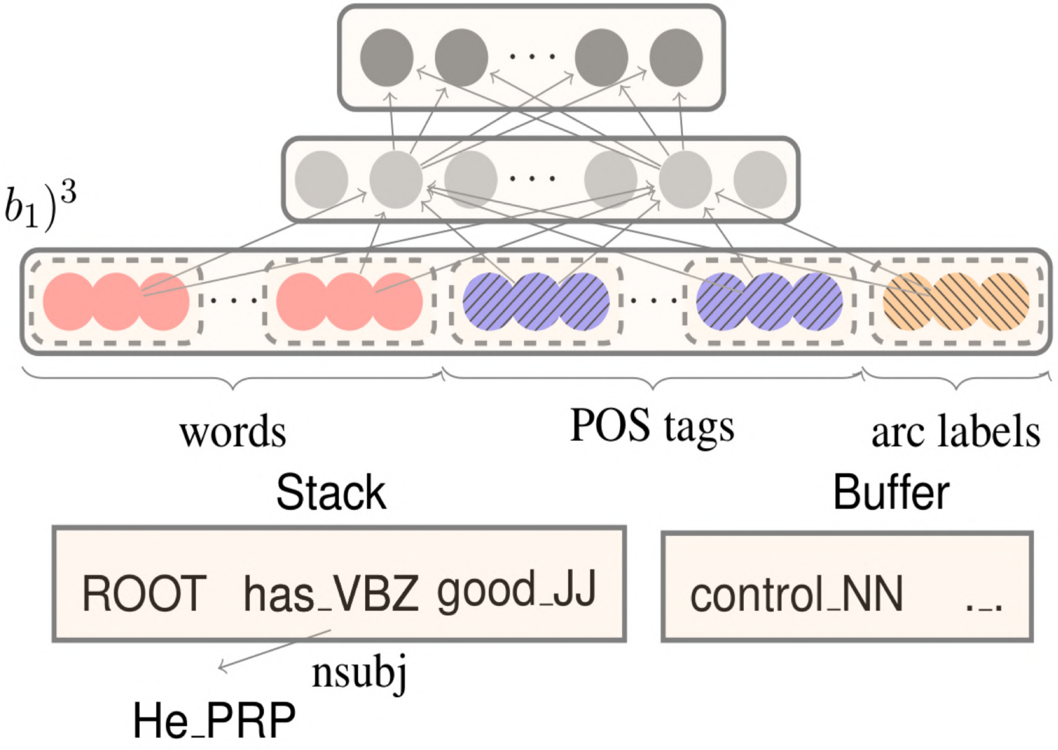
$$p = \mathtt{softmax}(W_2 h)$$

Hidden layer:

$$h = (W_1^w x^w + W_1^t x^t + W_1^l x^l + b_1)^3$$

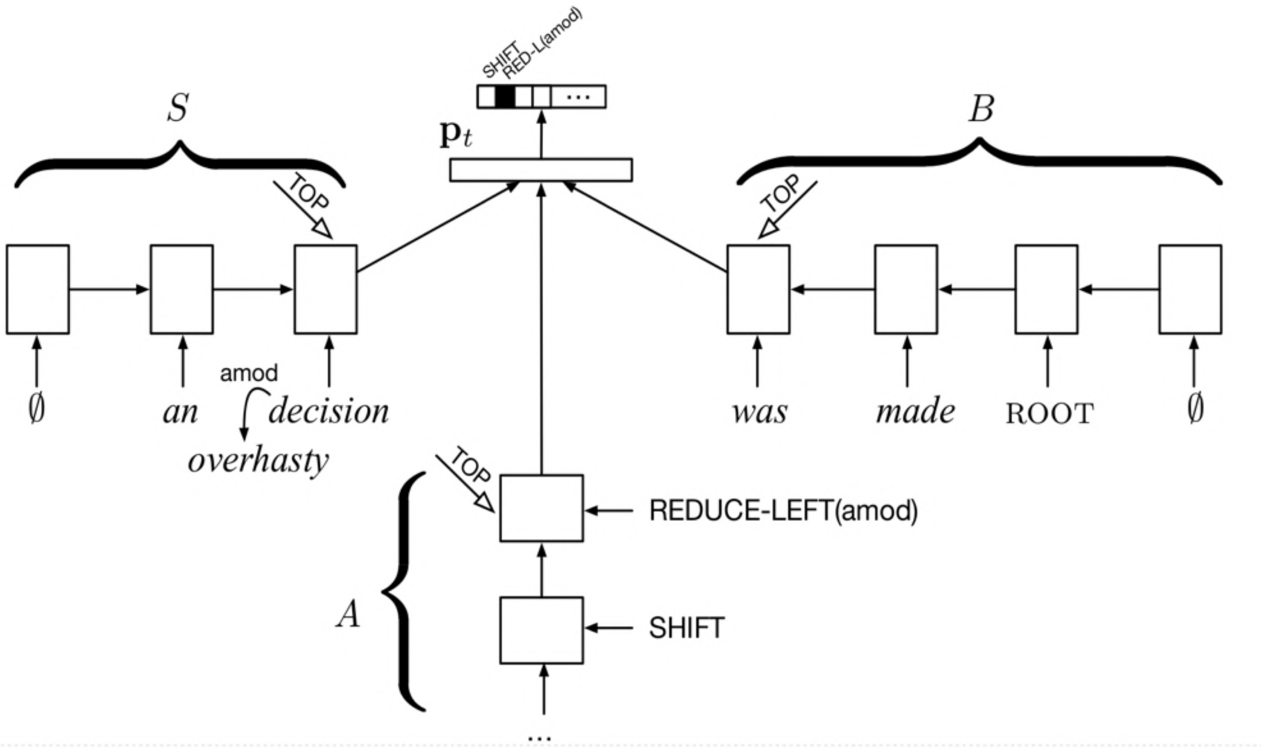
Input layer: $[x^w, x^t, x^l]$

Configuration



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

Stack-LSTM Parser



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

Sentence-level Log Likelihood

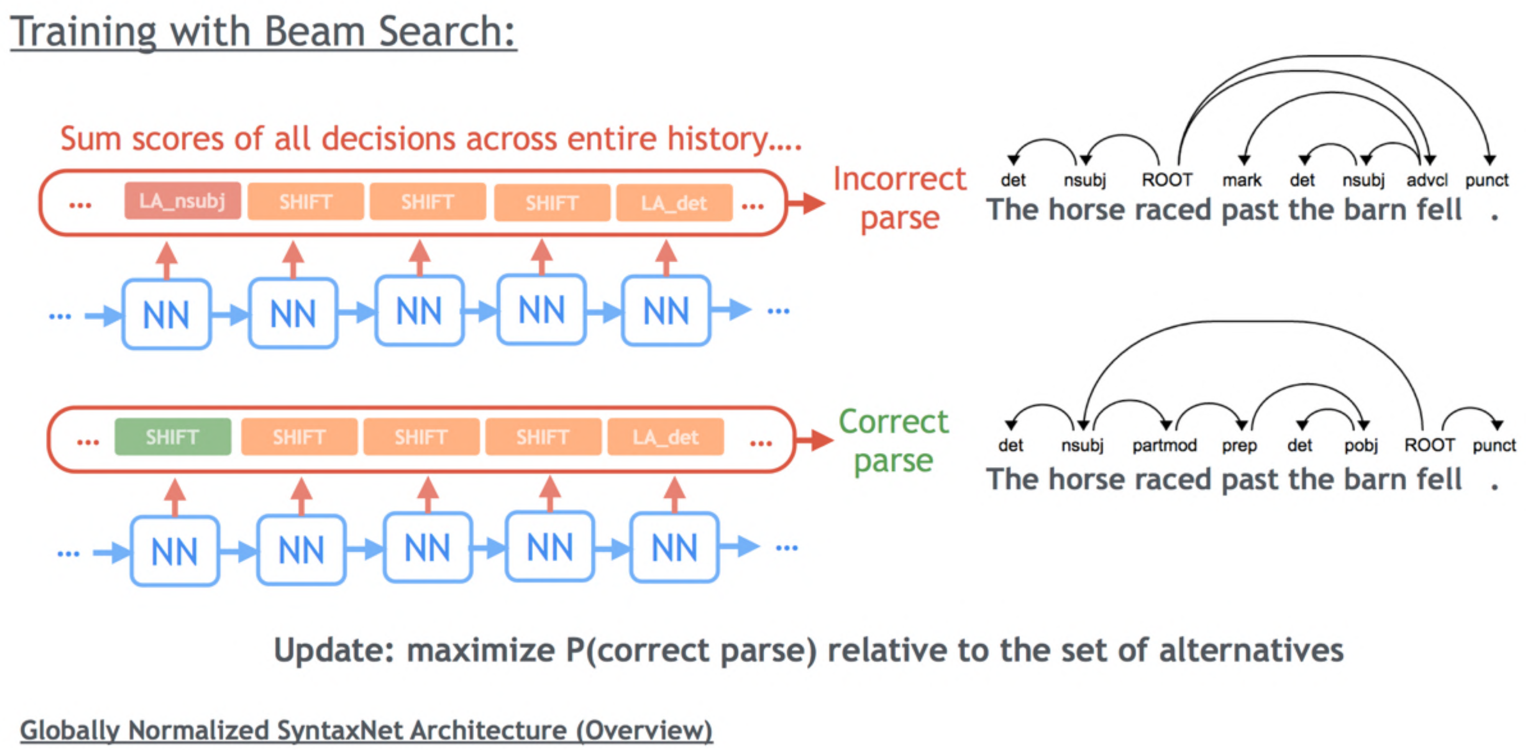
- Global Normalization
- Training with Beam Search

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

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

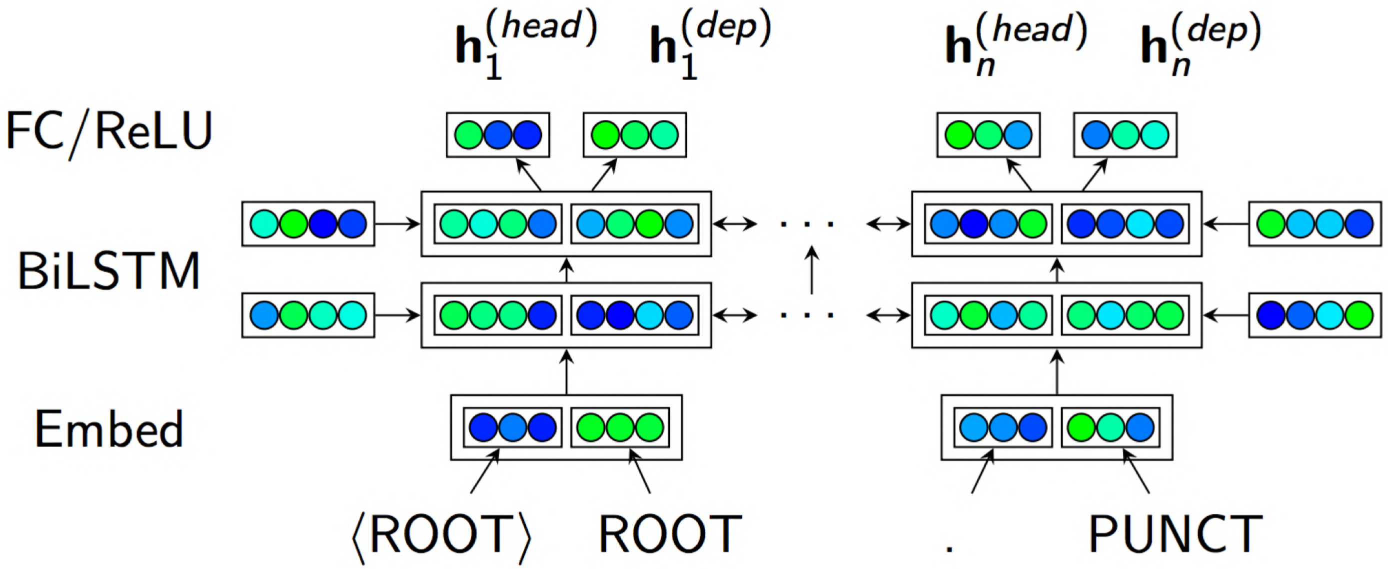
[Zhou, H., Zhang, Y., Huang, S., & Chen, J. A Neural Probabilistic Structured-Prediction Model for Transition-Based Dependency Parsing. ACL 2015]

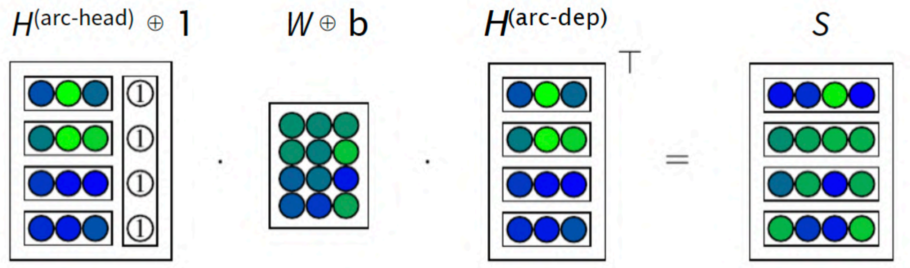
SyntaxNet: Google



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

Deep Biaffine Attention

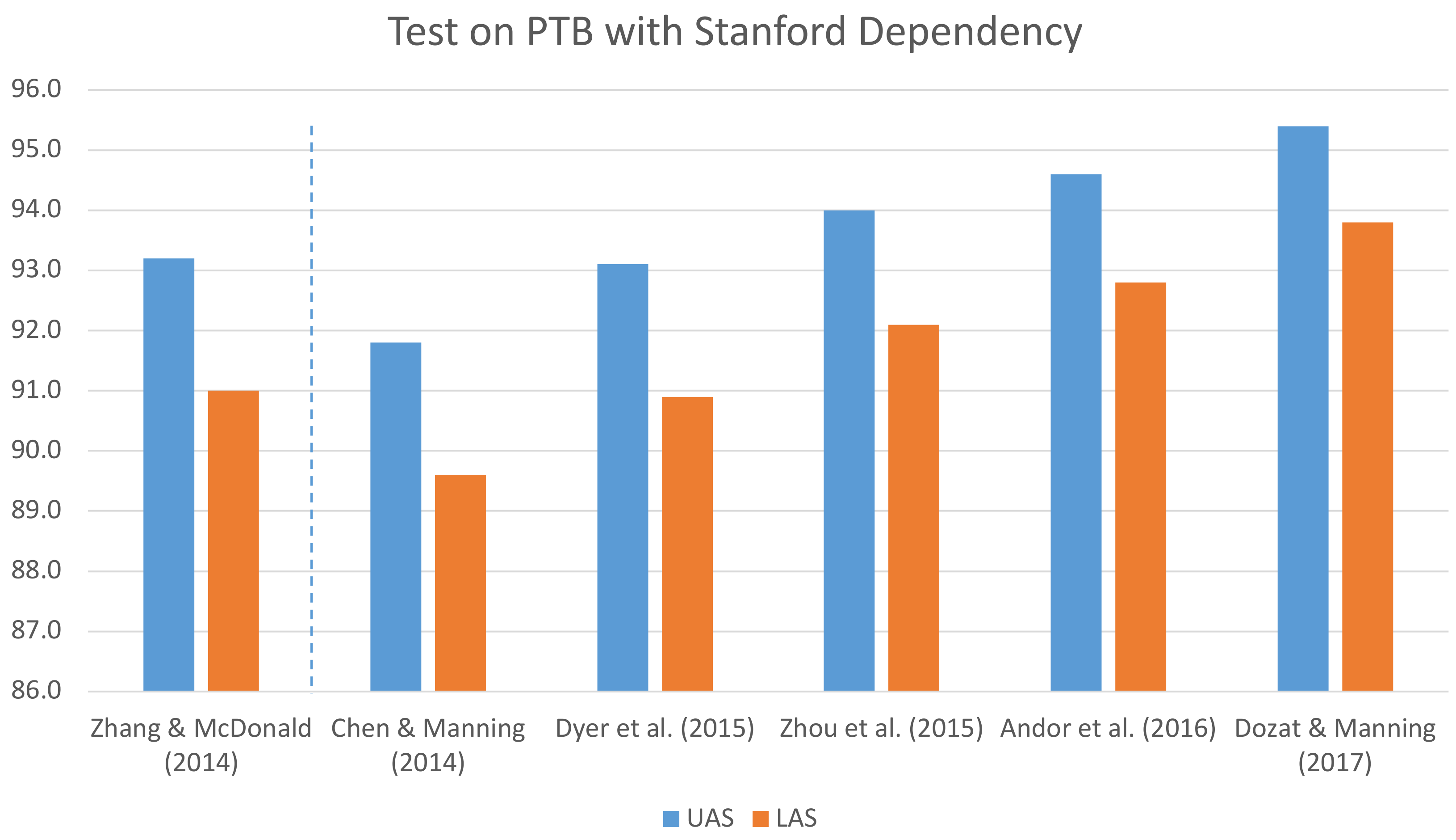




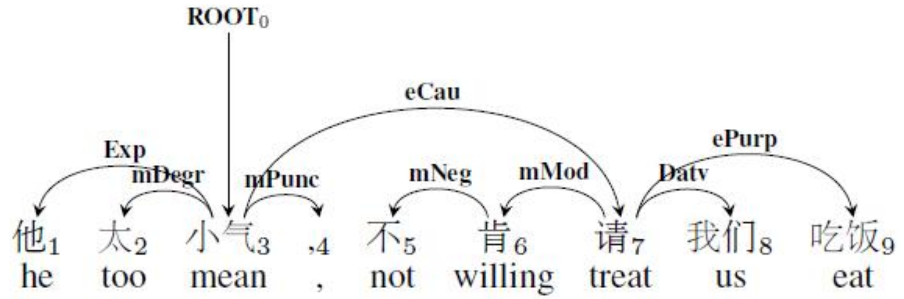
- Just optimize the likelihood of the head, no structured learning
- This is a local model, with global decoding using MST at the end

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

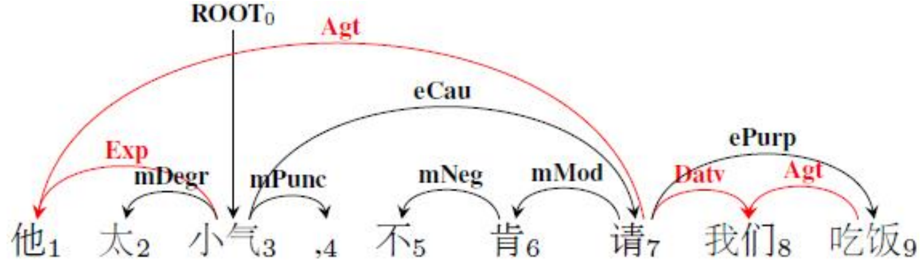
Changes of Performance



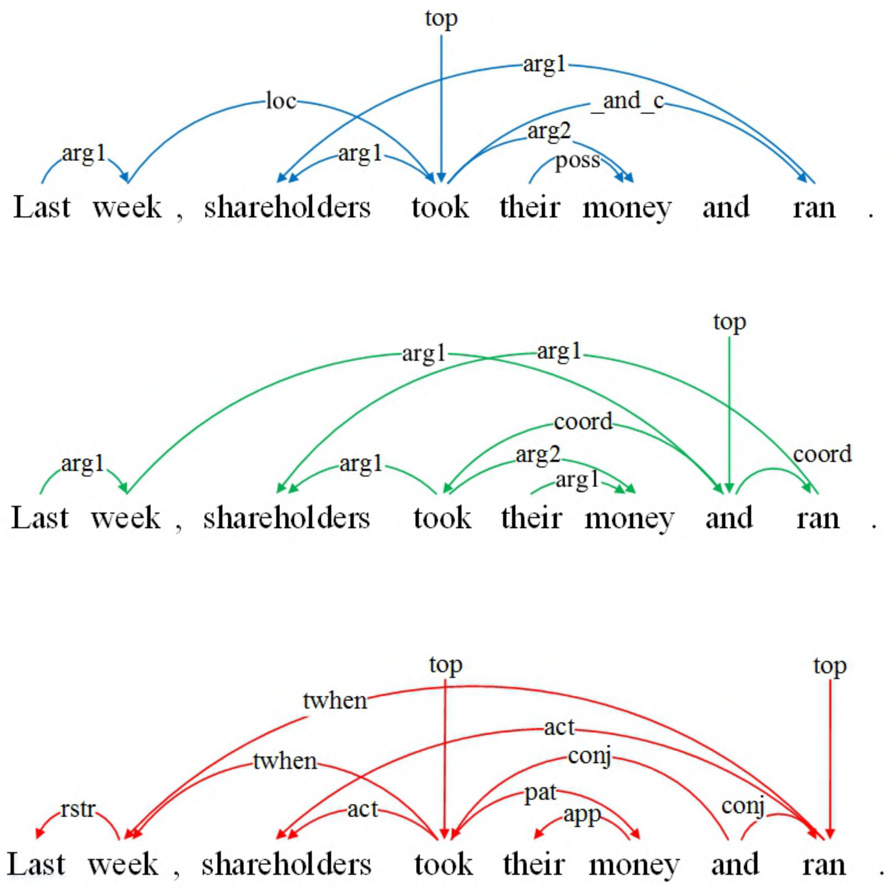
Semantic Dependency Graph Parser



SemEval 2012 Task 5 : Chinese Semantic Dependency (Tree)



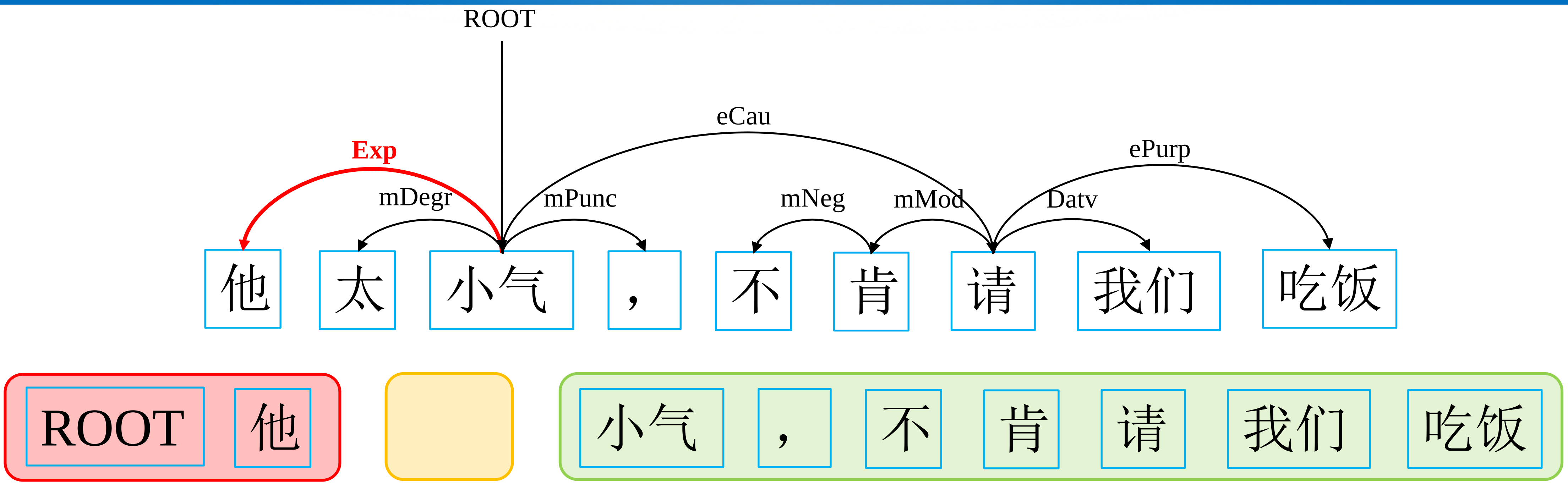
SemEval 2016 Task 9 : Chinese Semantic Dependency (Graph)



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

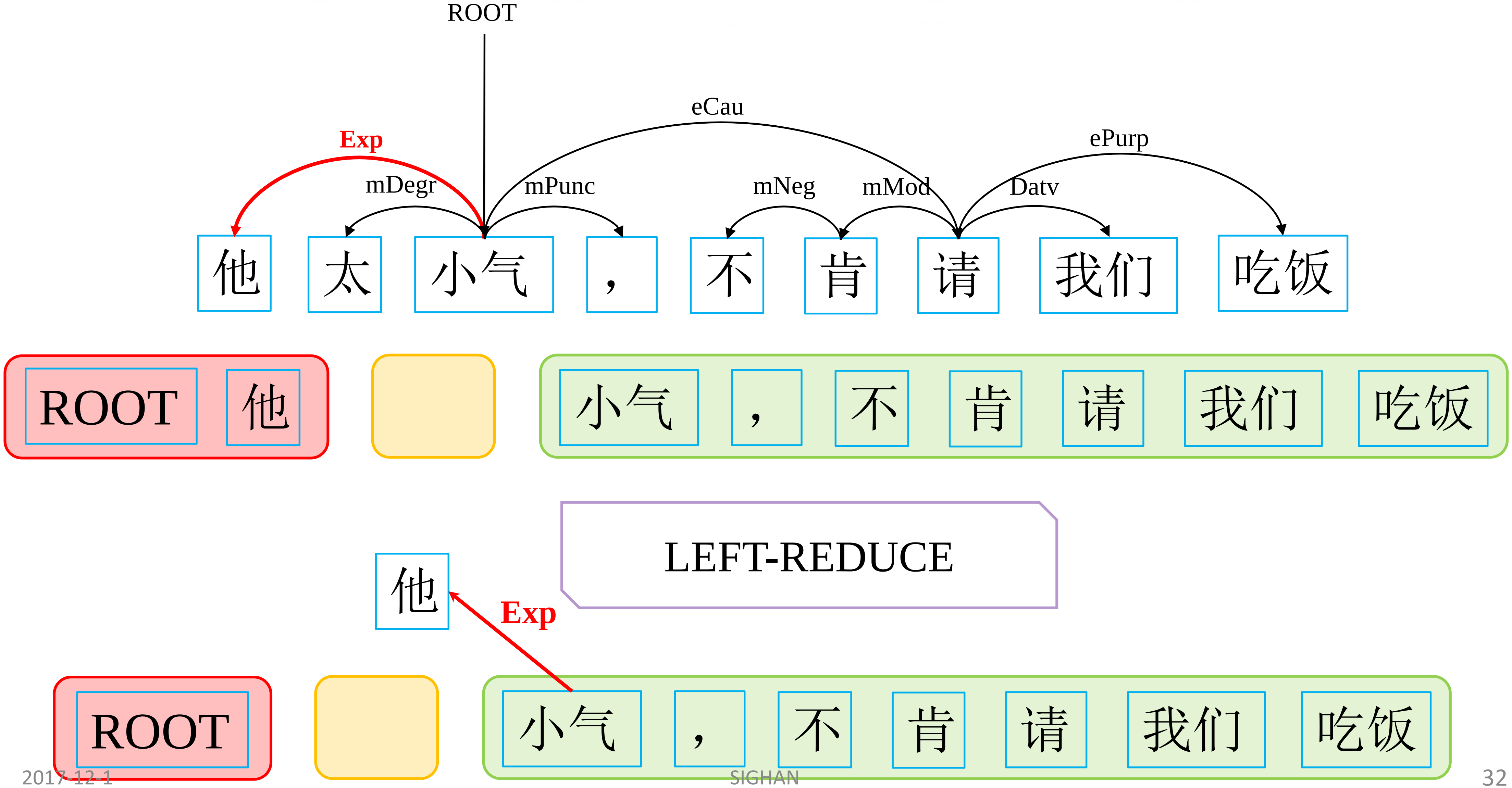
[Yuxuan Wang, Wanxiang Che, Jiang Guo and Ting Liu. A Neural Transition-Based Approach for Semantic Dependency Graph Parsing. AAAI 2018.]

Transition System for Dependency Tree [Choi and McCallum (2013)]

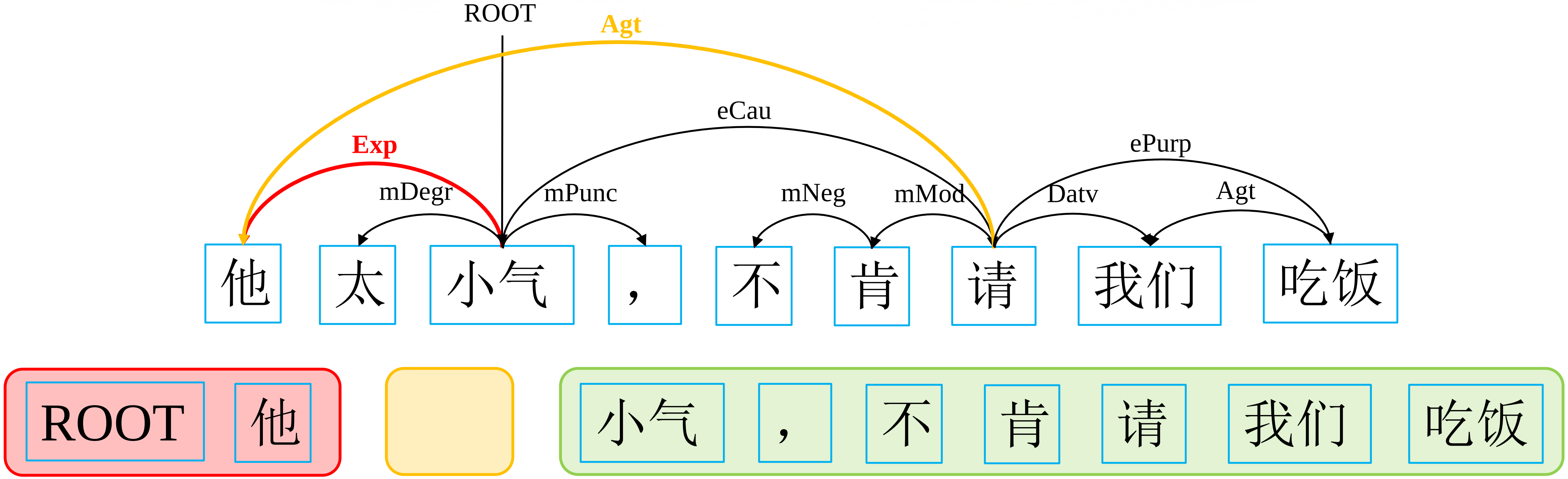


[Yuxuan Wang, Wanxiang Che, Jiang Guo and Ting Liu. A Neural Transition-Based Approach for Semantic Dependency Graph Parsing. AAAI 2018.]

Transition System for Dependency Tree [Choi and McCallum (2013)]

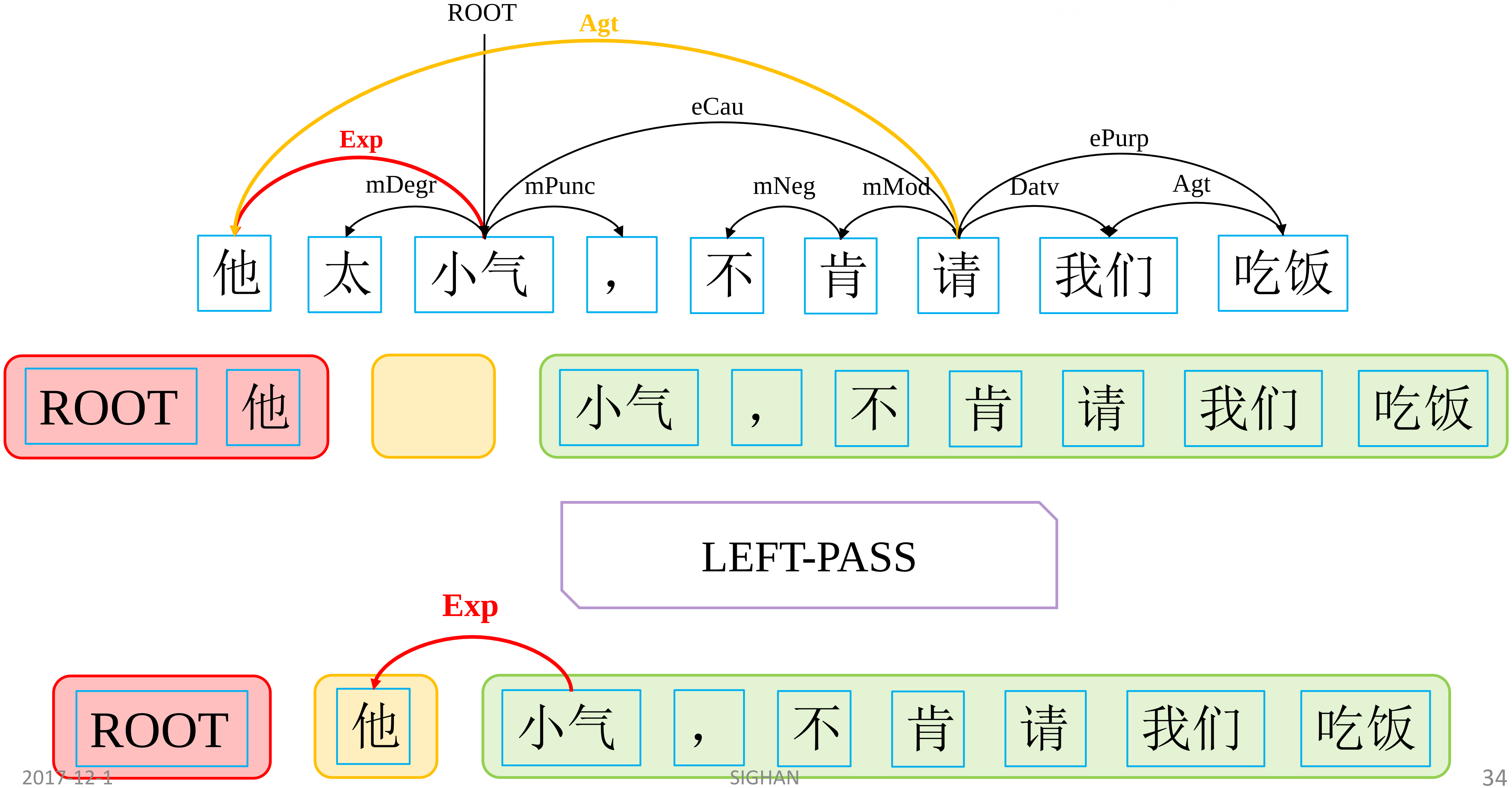


Transition System for Dependency Graph

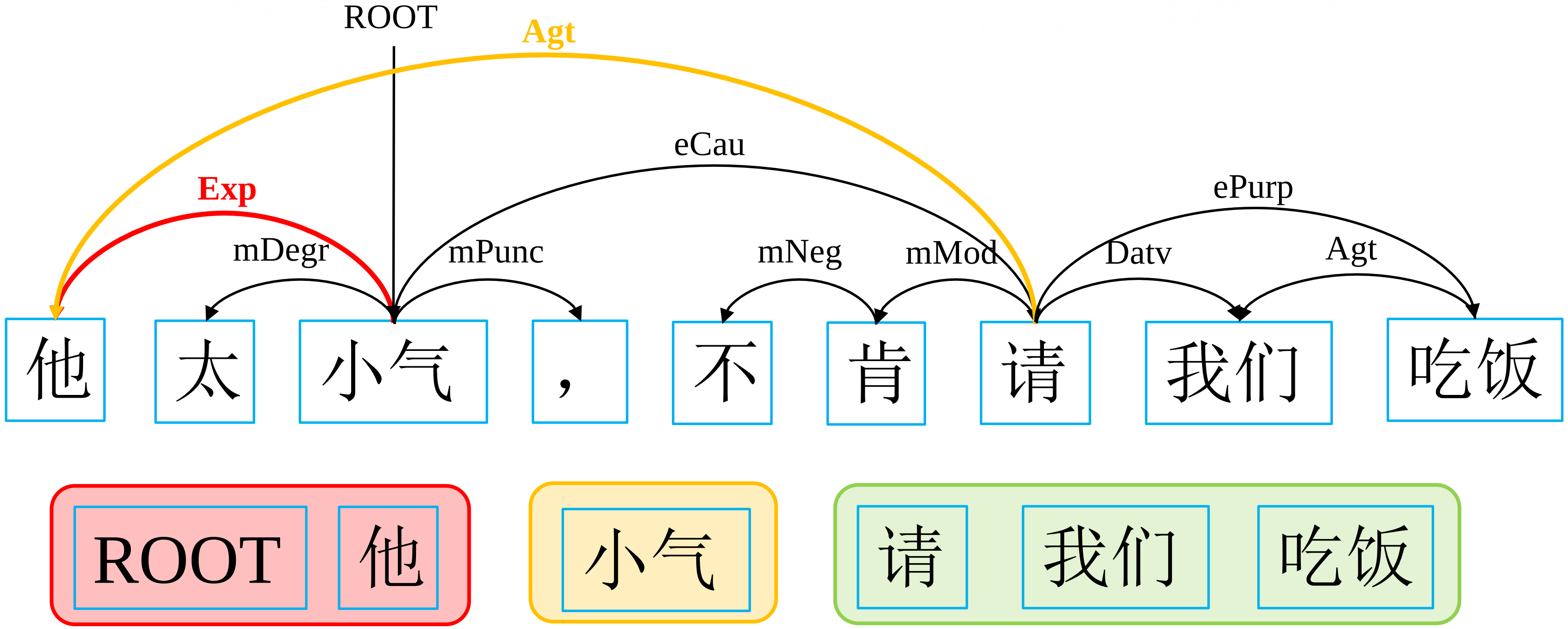


[Yuxuan Wang, Wanxiang Che, Jiang Guo and Ting Liu. A Neural Transition-Based Approach for Semantic Dependency Graph Parsing. AAAI 2018.]

Transition System for Dependency Graph

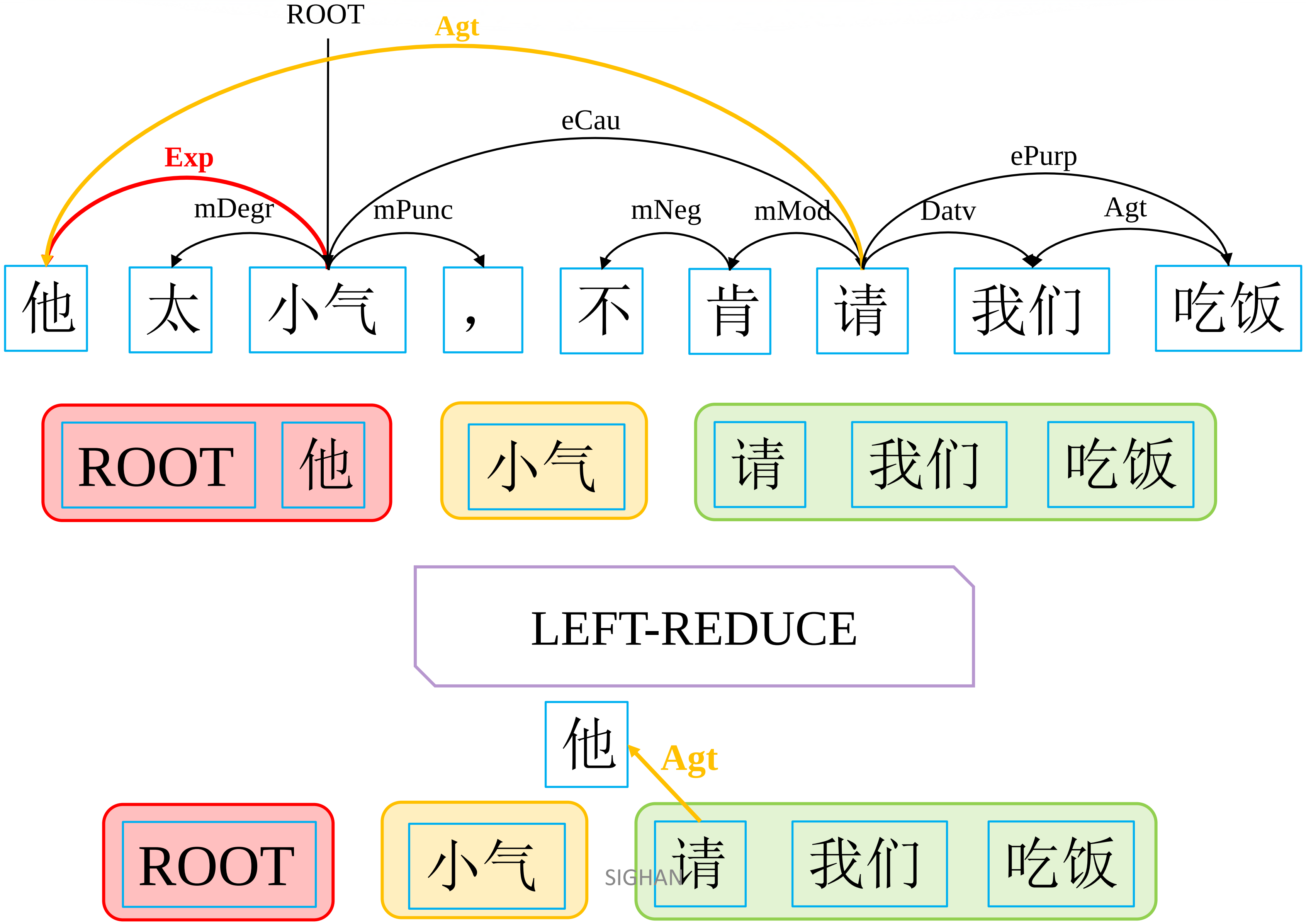


Transition System for Dependency Graph

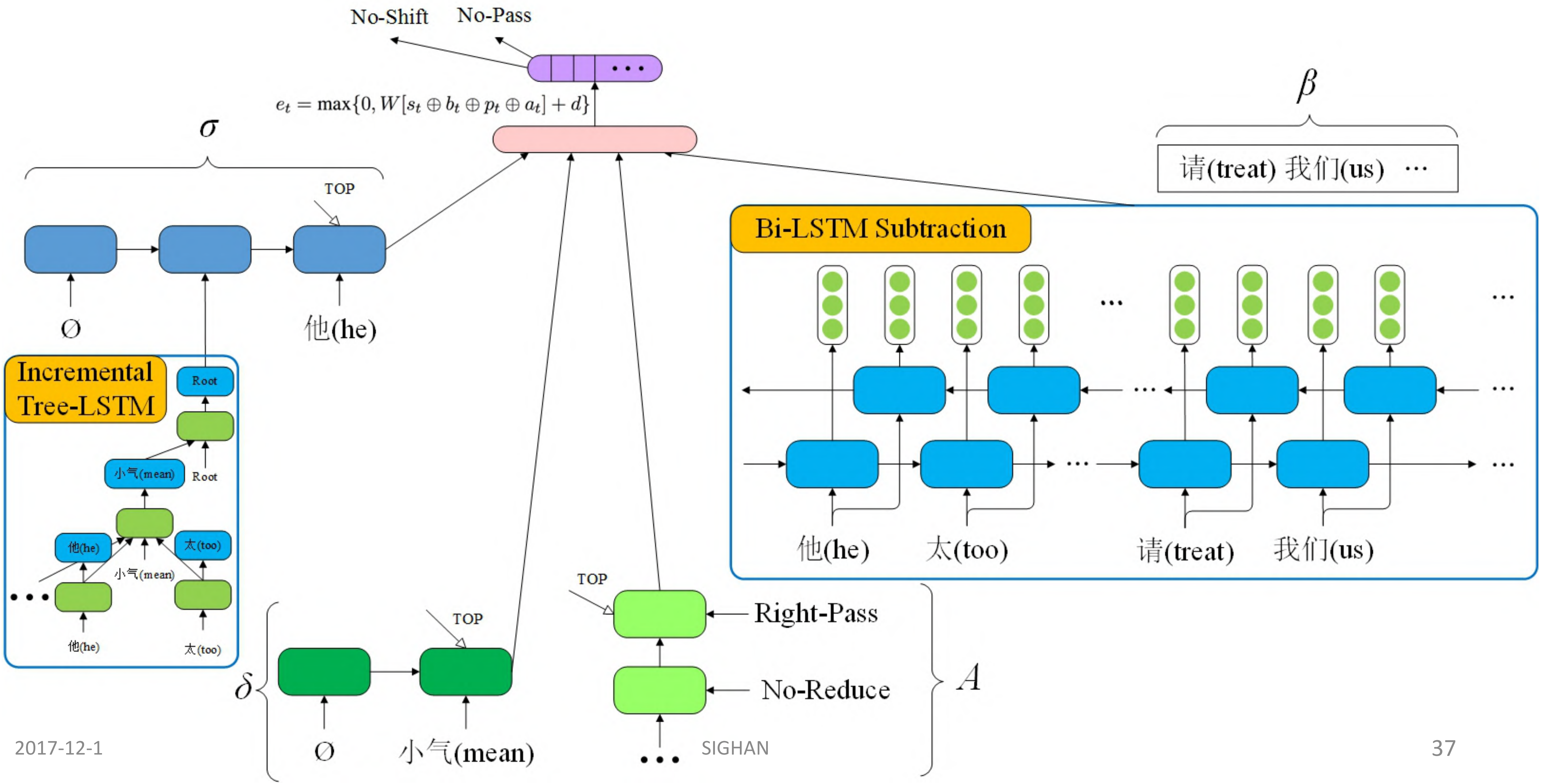


[Yuxuan Wang, Wanxiang Che, Jiang Guo and Ting Liu. A Neural Transition-Based Approach for Semantic Dependency Graph Parsing. AAAI 2018.]

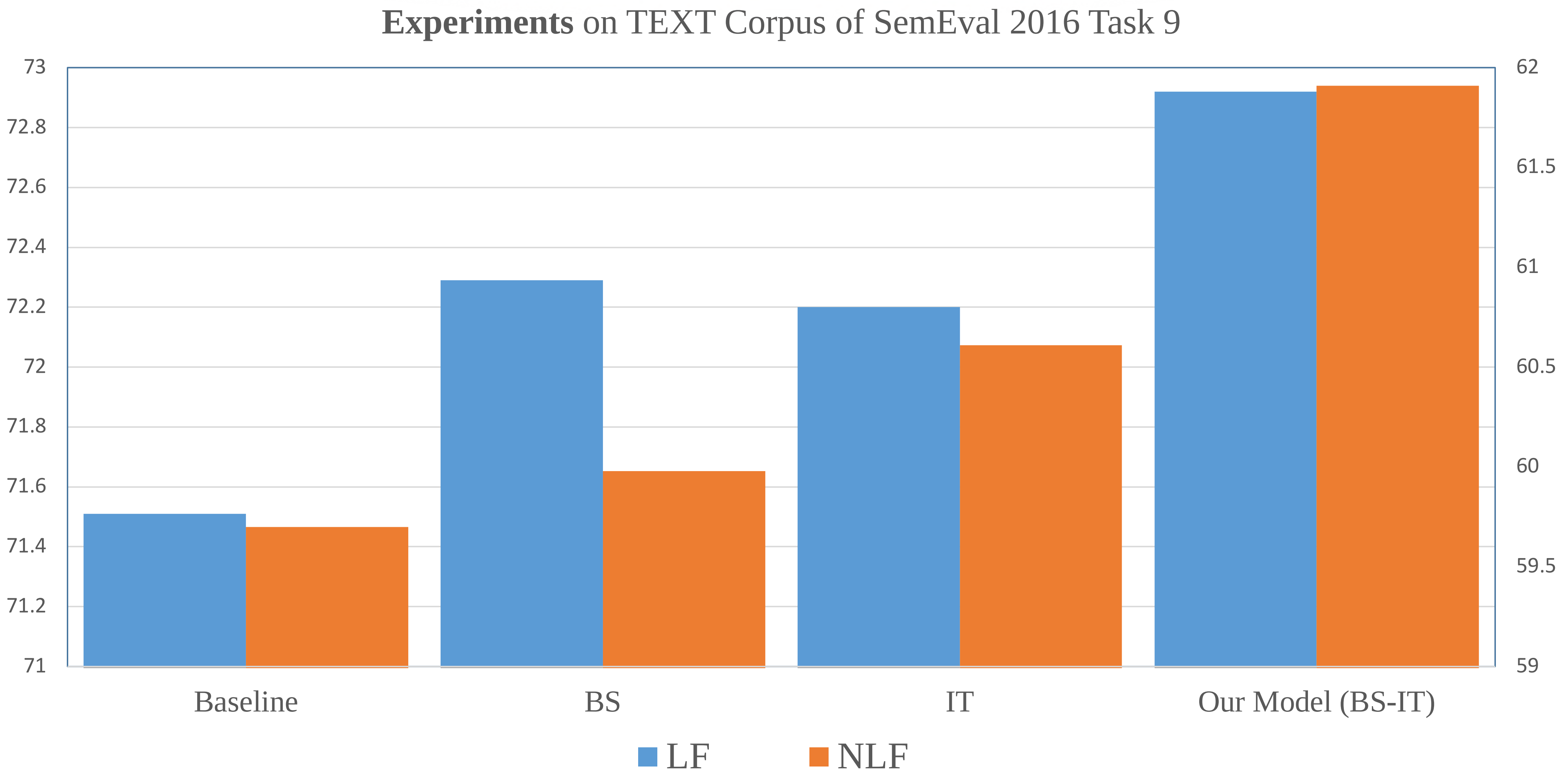
Transition System for Dependency Graph



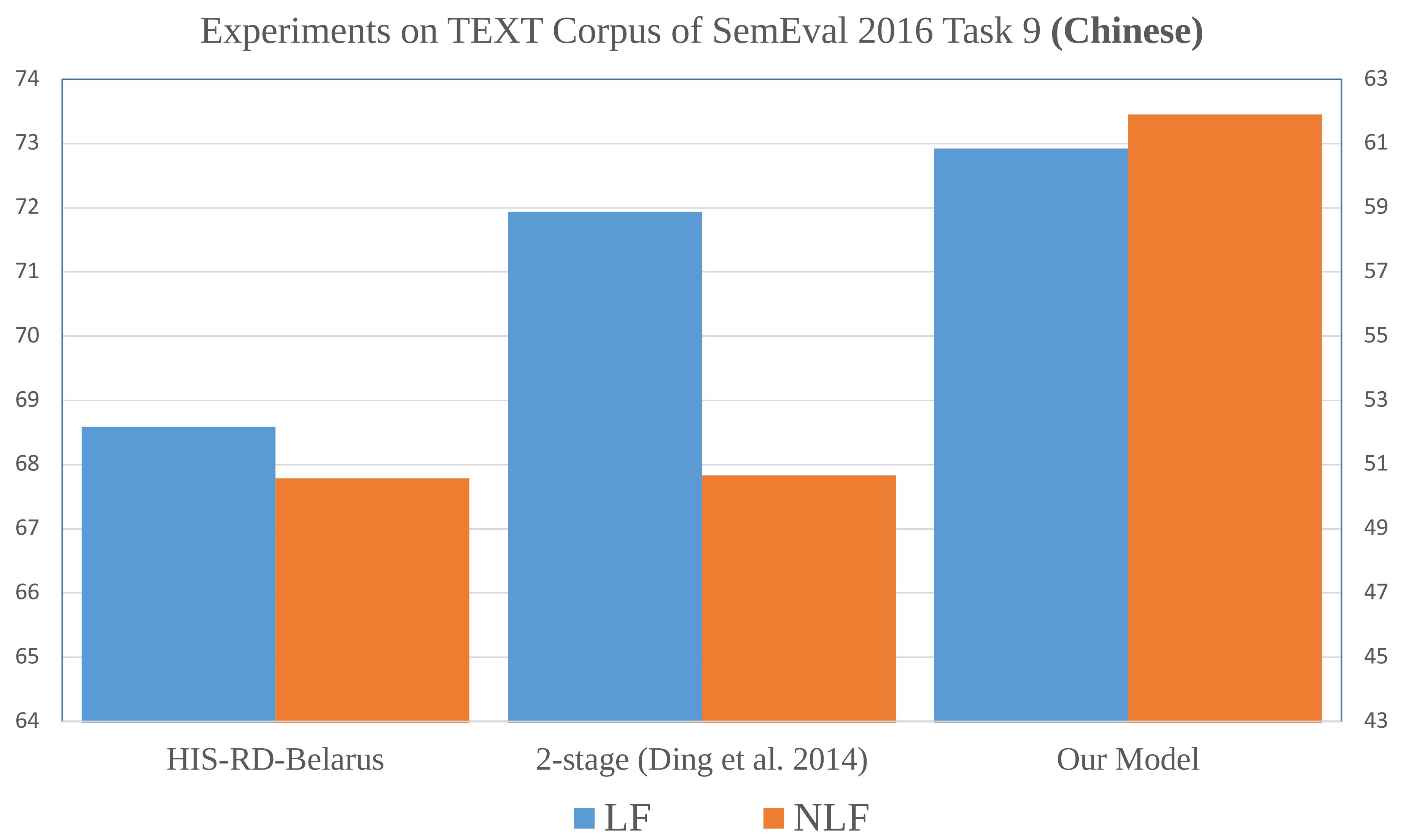
IT-BS Classifier



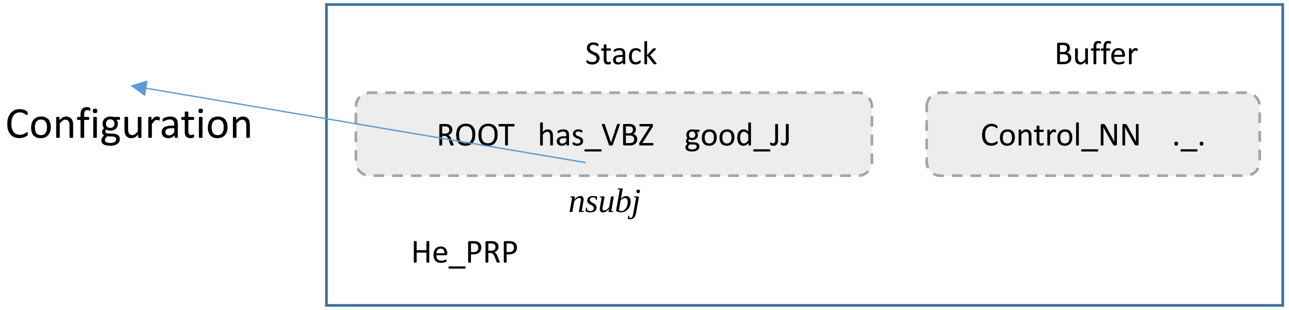
Experiments



Experiments



DL for NLP: End-to-End Learning

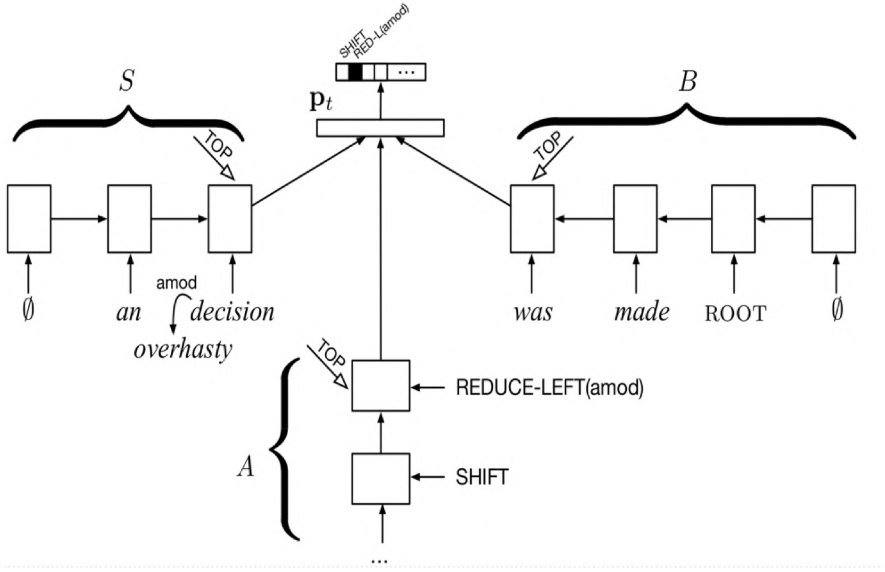


from single words
S_0wp; S_0w; S_0p; N_0wp; N_0w; N_0p; N_1wp; N_1w; N_1p; N_2wp; N_2w; N_2p;
from word pairs
S_0wpN_0wp; S_0wpN_0w; S_0wN_0wp; S_0wpN_0p; S_0pN_0wp; S_0wN_0w; S_0pN_0p N_0pN_1p
from three words
$N_0pN_1pN_2p$; $S_0pN_0pN_1p$; $S_{0h}pS_0pN_0p$; $S_0pS_{0l}pN_0p$; $S_0pS_{0r}pN_0p$; $S_0pN_0pN_{0l}p$

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;

0	0	1	0	1	…	0	1	0	0

Traditional Parser

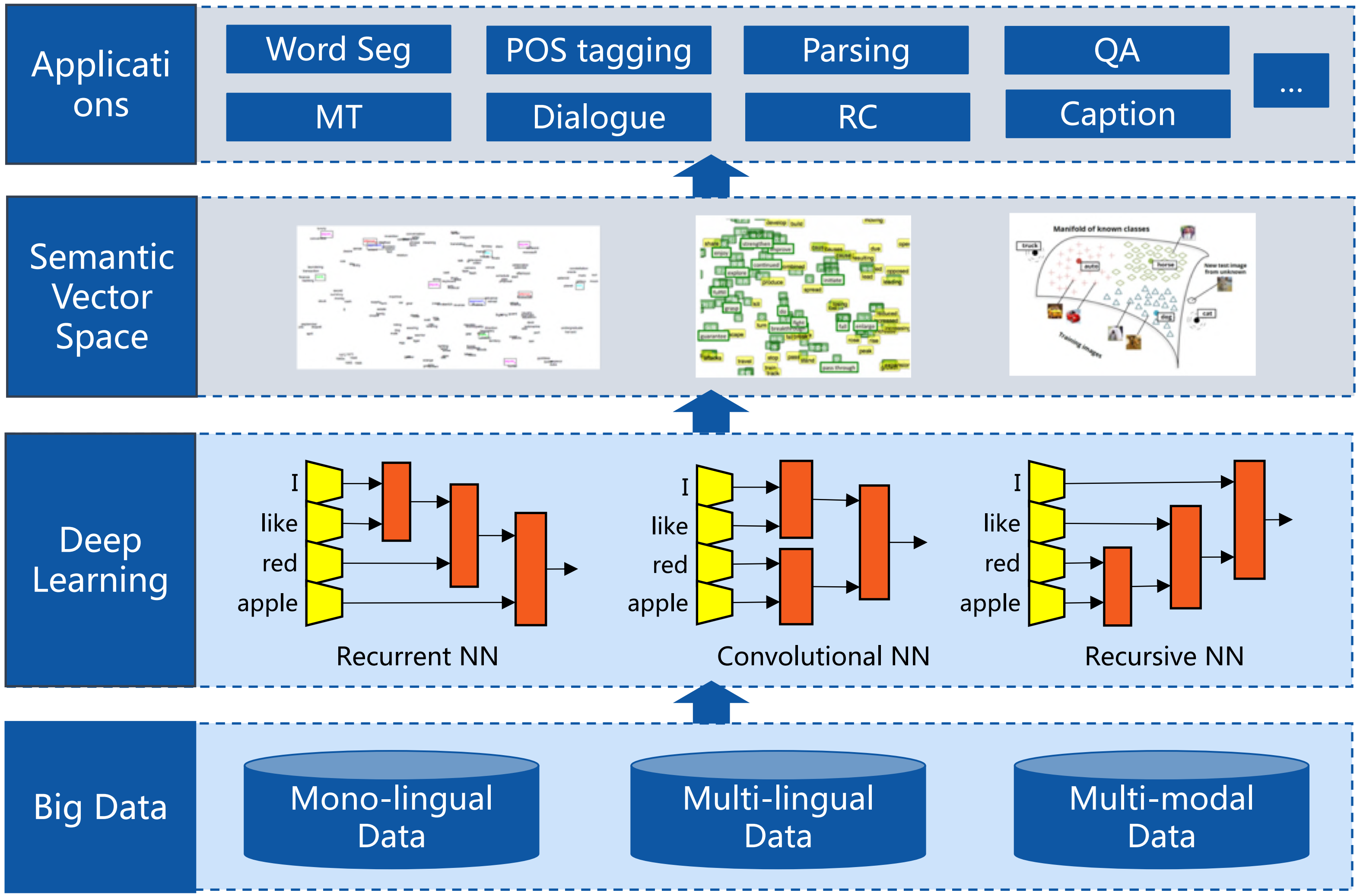


Stack-LSTM Parser

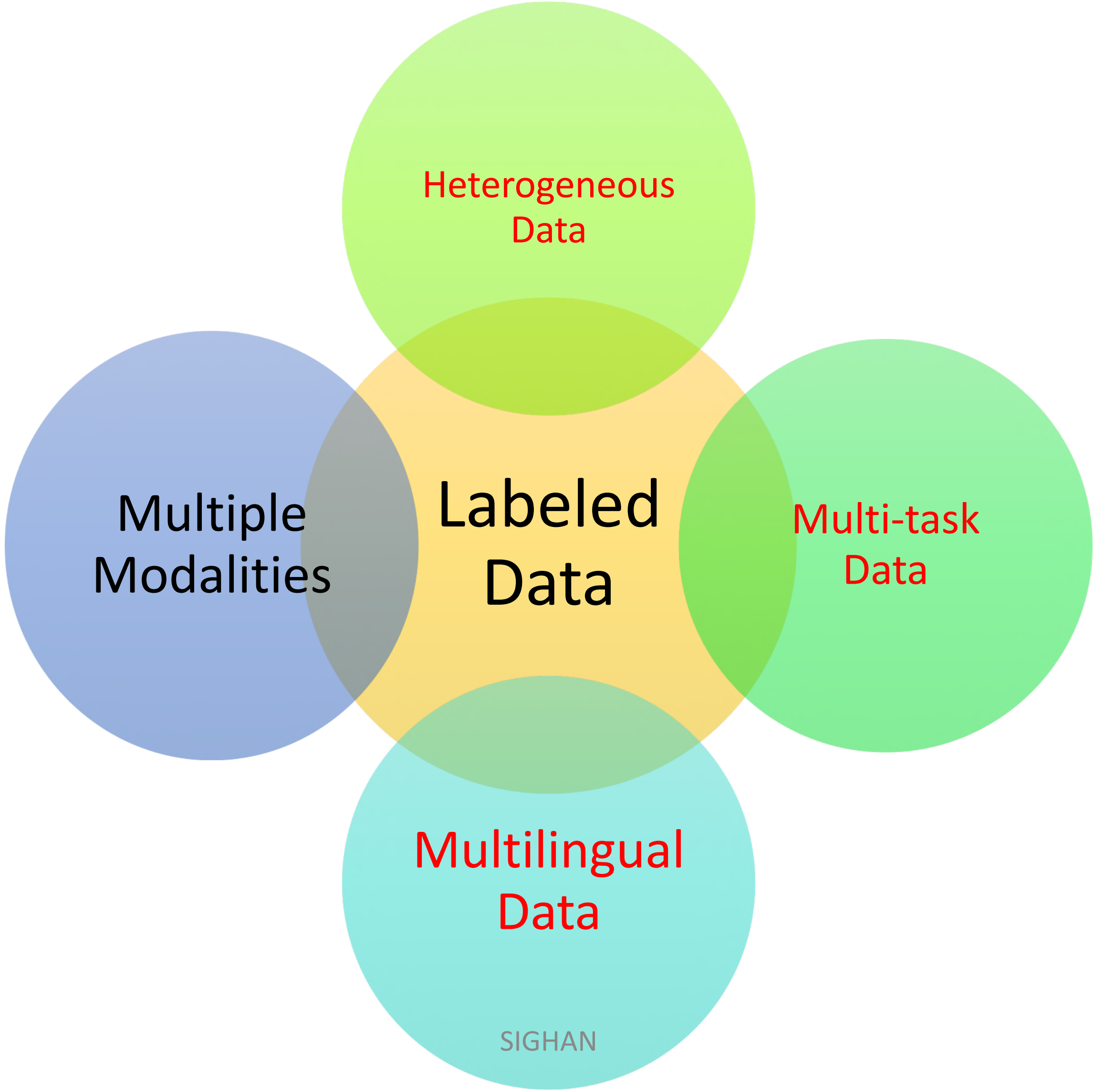
Outline

1. Syntactic and Semantic Parsing
2. Pseudo Data for Parsing
3. Applications of Parsing
4. Summary

DL for NLP: Representation Learning

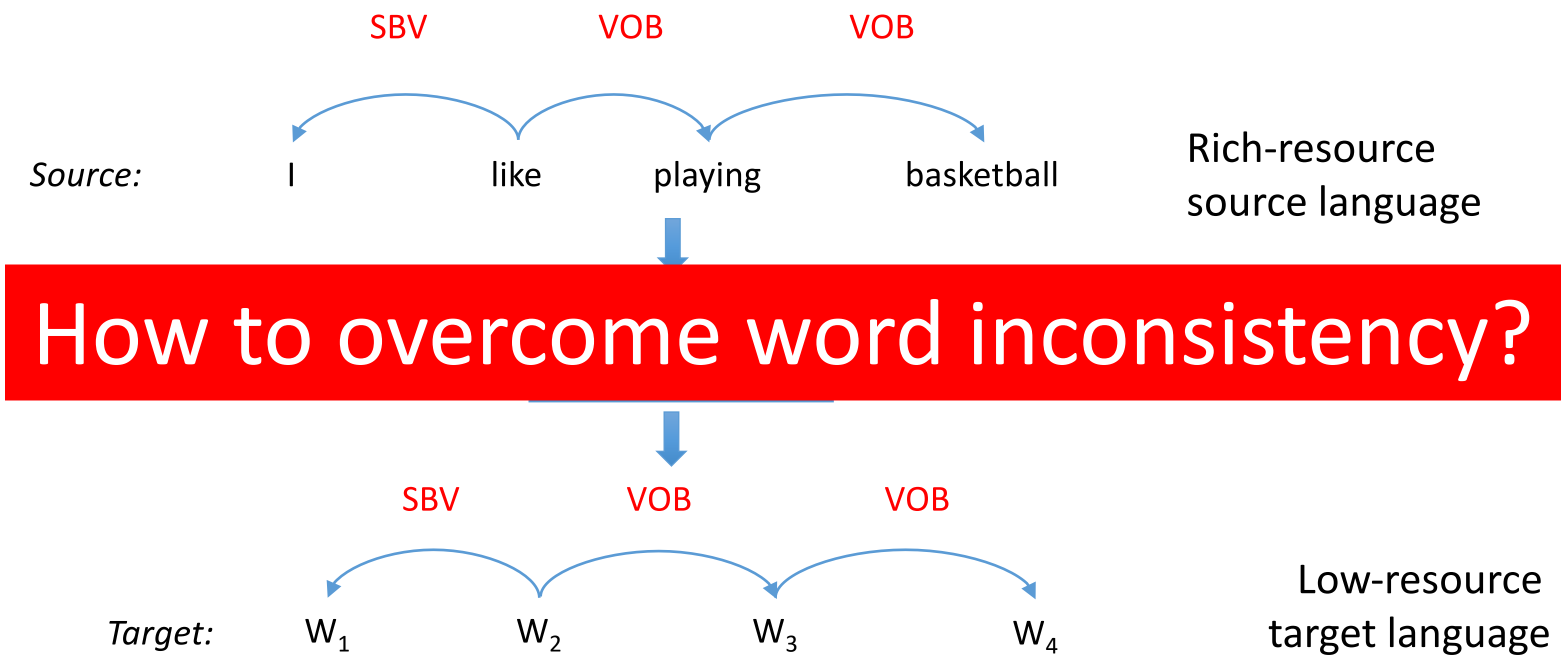


Pseudo Data for Parsing



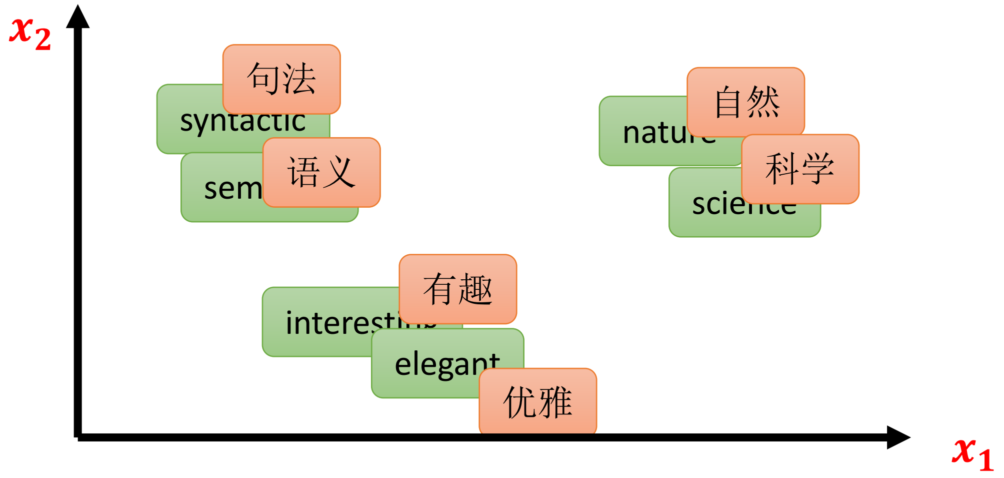
Cross-language Parser

- Transfer the parser trained on source language(s) to parse a target language



Cross-language Parser

- Learn bilingual word embeddings to overcome word inconsistency



Published papers: ACL 2015， AAAI 2016，JAIR 2016，CoNLL 2017

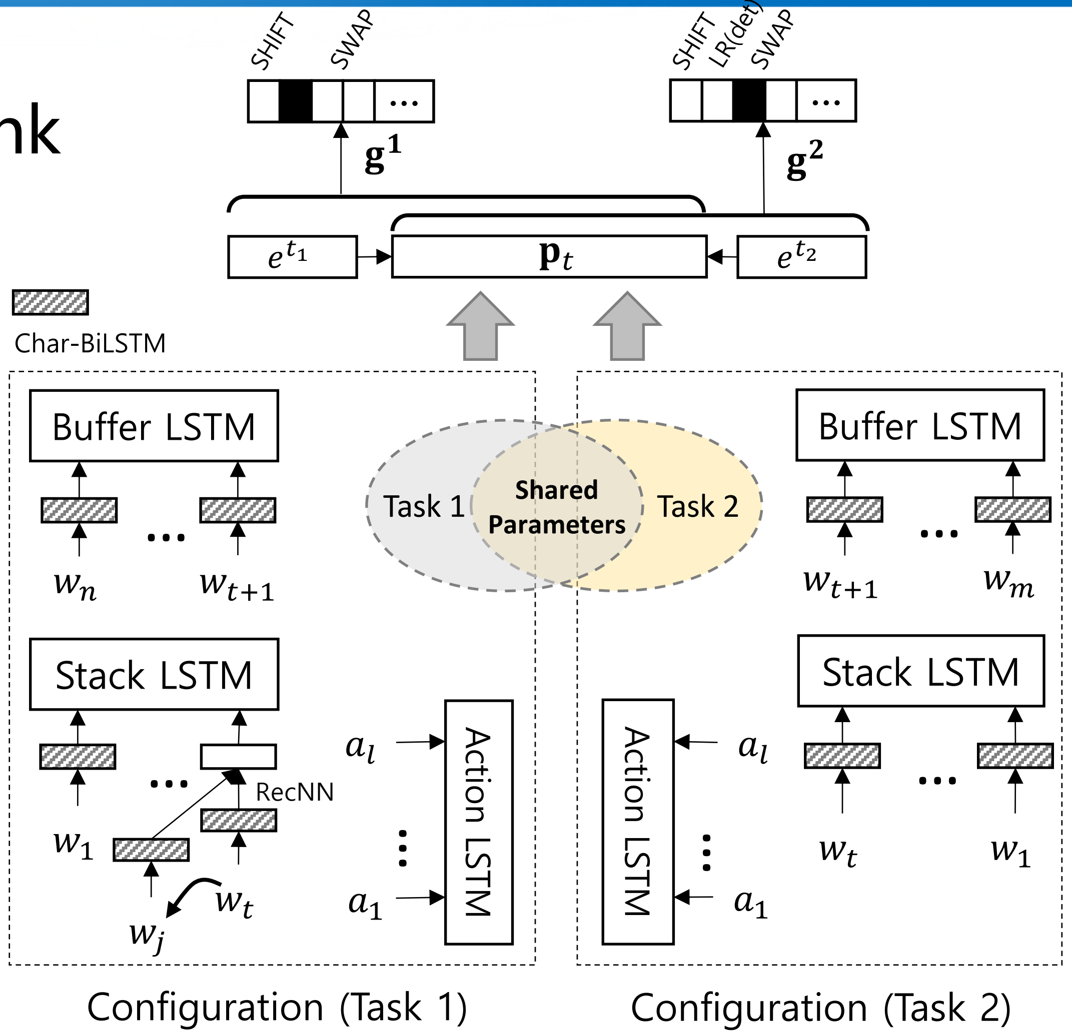
Deep Multi-Task Learning Architecture

Each task corresponds to a Treebank

- Multilingual universal
- Monolingual heterogeneous
- Multiple NLP tasks

Core Parameters

- LSTM(B), LSTM(S)
- LSTM(A)
- BiLSTM(chars)
- RecNN
- W_A, W_B, W_S
- $E_{pos}, E_{char}, E_{rel}, E_{act}$
- e^t
- $\mathbf{g}$



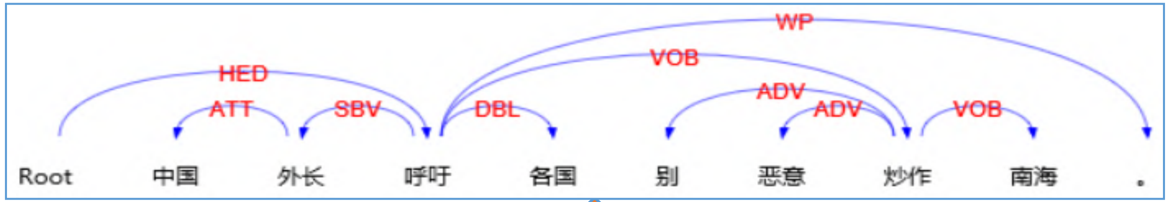
[Jiang Guo, Wanxiang Che, Haifeng Wang and Ting Liu. A Universal Framework for Transfer Parsing across Multi-typed Treebanks. Coling 2016]

Outline

1. Syntactic and Semantic Parsing
2. Pseudo Data for Parsing
3. Applications of Parsing
4. Summary

Shallow Learning

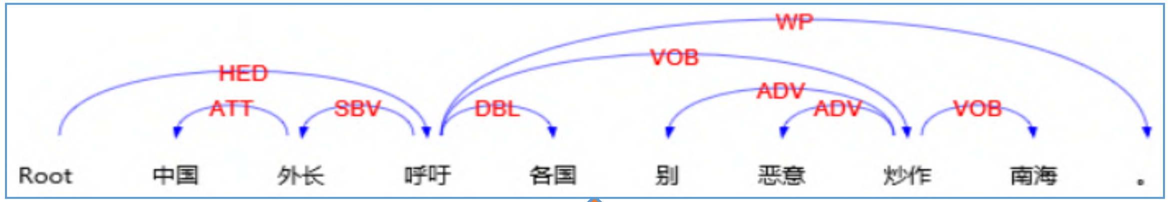
The final task, e.g., entity relation extraction



Sentence

Deep Learning

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



End-to-End

Sentence

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]

Language Technology Platform (LTP)

- http://ltp.ai
- Rich and accurate Chinese 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
  - 4th place/33/113 at CoNLL 2017: multilingual syntactic dependency parsing

LTP Demo

Not Secure | ltp.ai/demo.html

LTP

主页 新闻 在线文档 在线演示 全部下载 常见问题 Github 语言云

在线演示

国务院总理李克强调研上海外高桥时提出，支持上海积极探索新机制。

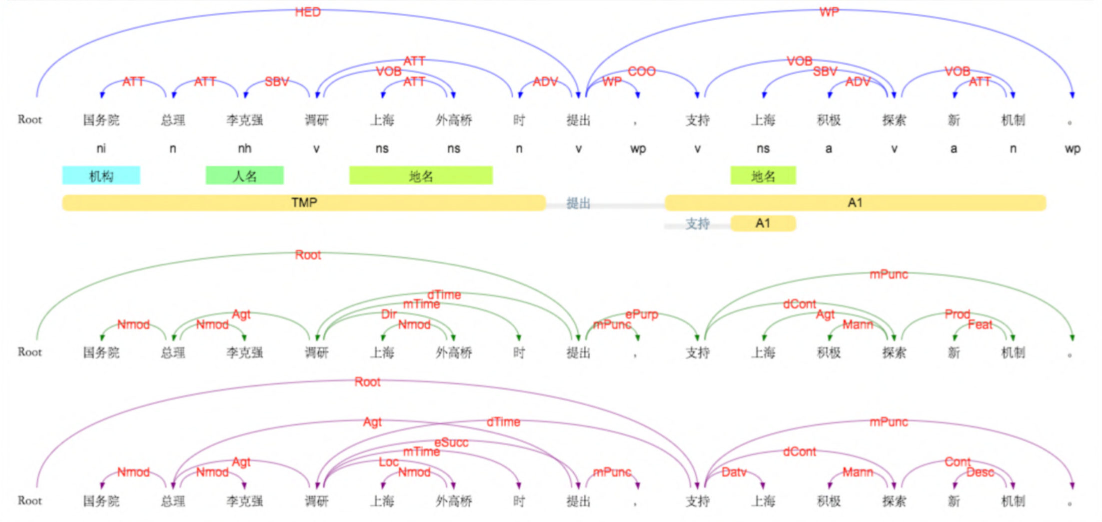
样例

分析

句子视图 篇章视图 XML视图

☑ 词性标注 ☑ 命名实体 ☑ 句法分析（蓝） ☑ 语义角色标注 ☑ 语义依存（树）分析（绿） ☑ 语义依存（图）分析（紫）

段落1句子1:国务院总理李克强调研上海外高桥时提出，支持上海积极探索新机制。



标签释义

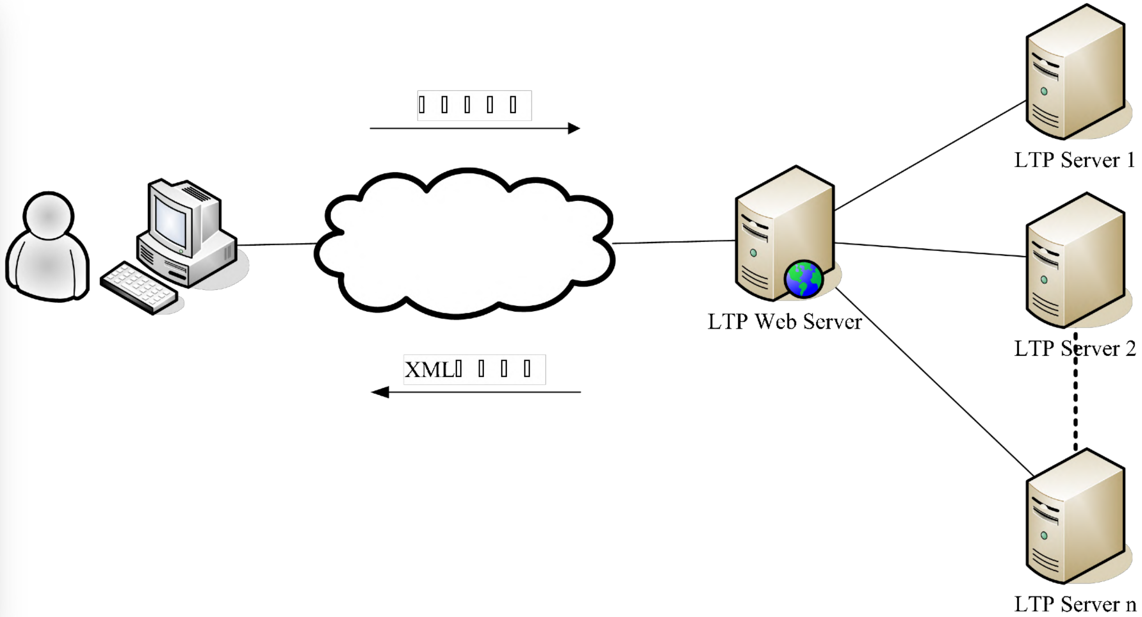
语义依存分析 依存句法分析 词性标注 语义角色标注

Tag	关系类型	Description
Agt	施事关系	Agent
Cont	客事关系	Content
Datv	源事关系	Dative
Desc	描写角色	Description
Dir	趋向角色	Direction
ePurp	目的关系	event Purpose
eSucc	顺承关系	event Successor
Feat	描写角色	Description
Loc	空间角色	Location
Mann	方式角色	Manner
mPunc	标点标记	Punctuation Marker
mTime	时间标记	Time
Nmod	名字修饰角色	Name-modifier
Prod	成事关系	Product
Root	根节点	Root

在文档中查看全部标签信息

LTP-Cloud Service

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



LTP-Cloud API Example

```
import urllib2, urllib, sys
uri_base = http://api.ltp-cloud.com/analysis/?
api_key  = "YourAPIKey"
text     = urllib.quote("我爱北京天安门")
format   = sys.argv[1]
url = "{}api_key={}&text={}&format={}&pattern=all".format(uri_base, api_key, text, format)
print urllib2.urlopen(url).read()
```

More Documents

- https://github.com/HIT-SCIR/ltp-cloud-api-tutorial

How to Use Tree or Graph Structures?

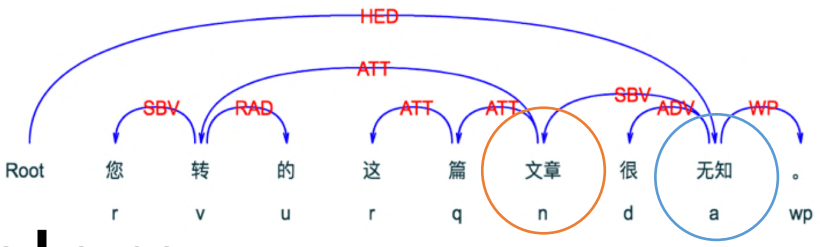
- As Information Extraction Rules
- As Input Features
- As Input Structures
- As Structured Prediction

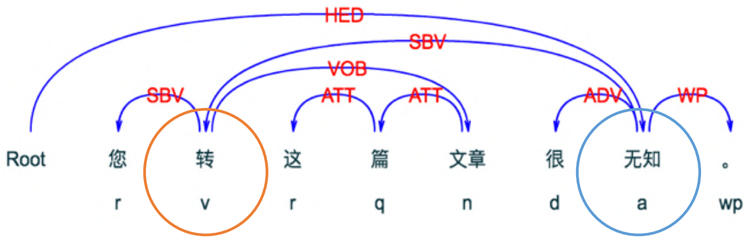
How to Use Tree or Graph Structures?

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

As Information Extraction Rules

- For example
  - Polarity-target pair extraction

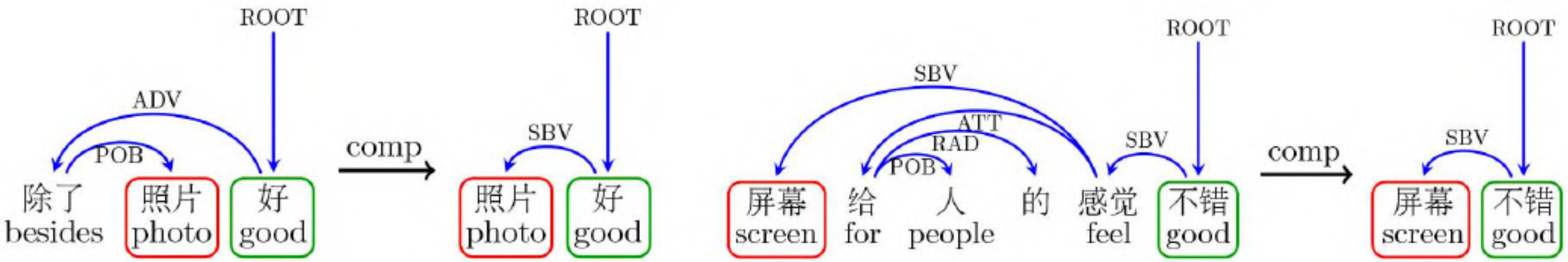




- Problem
  - The extraction rules are very complex
  - The parsing results are inexact

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%

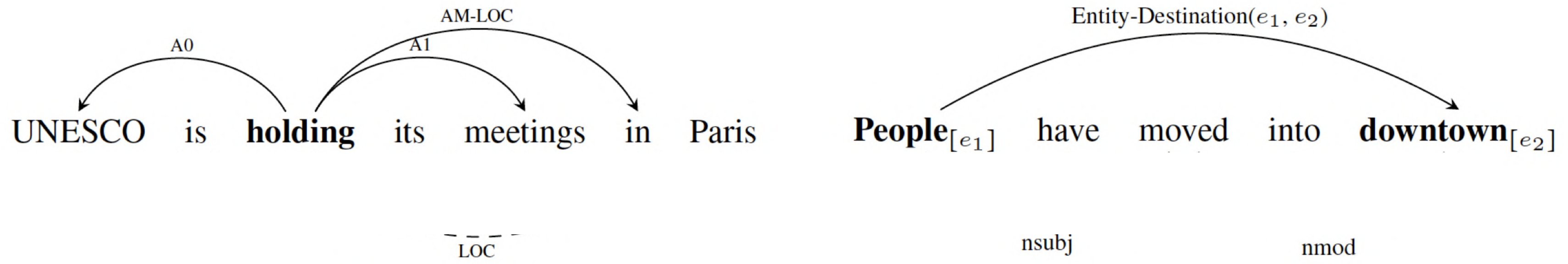
[Wanxiang Che, Yanyan Zhao, Honglei Guo, Zhong Su, Ting Liu. Sentence Compression for Aspect-Based Sentiment Analysis. IEEE/ACM Transactions on Audio, Speech, and Language Processing. 2015, 23(12)]

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)

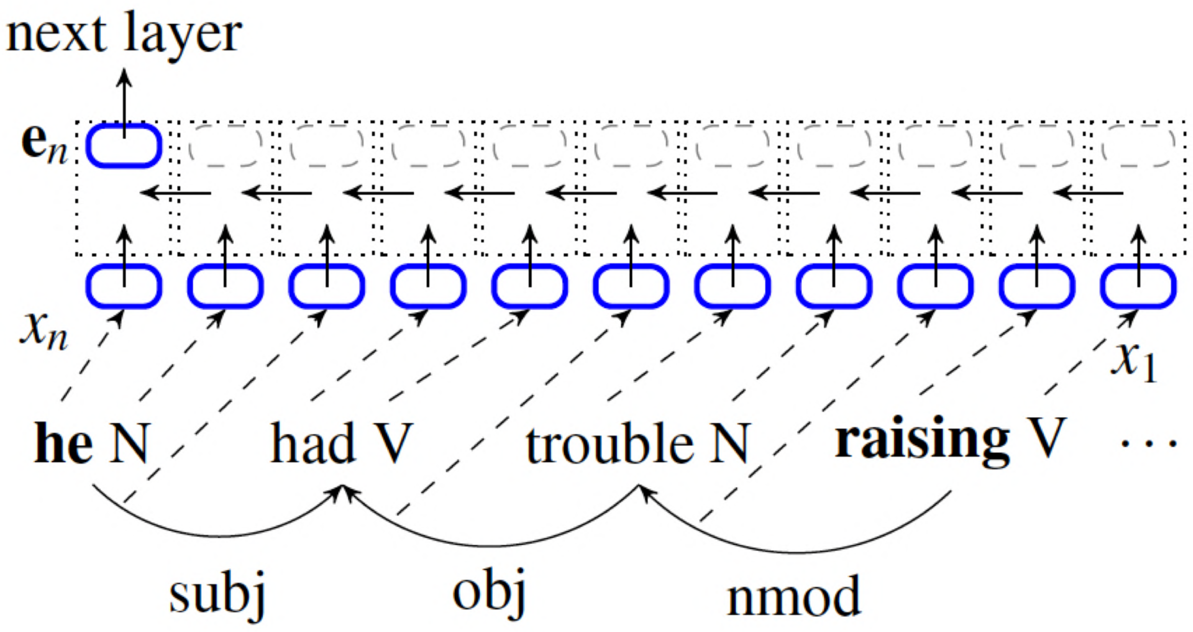


(a) Semantic Role Labeling.　(b) Relation Classification.

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

Path Features

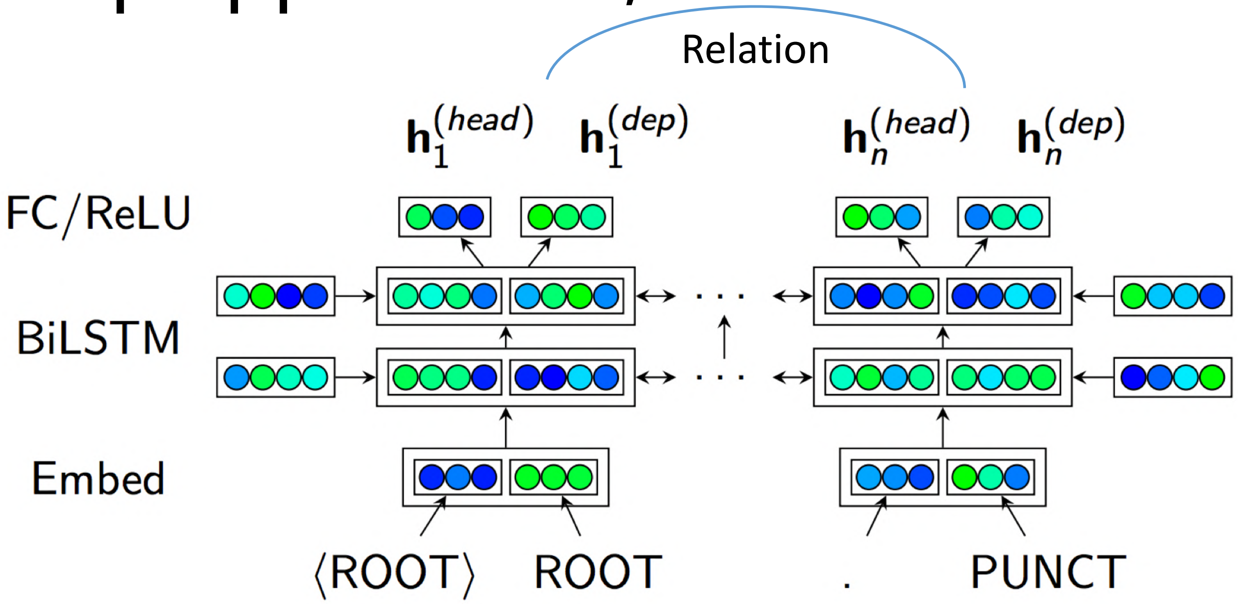
- Use LSTMs to represent paths
- All of word, POS tags and relations can be inputted



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

Hidden Units of Parsing as Features

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



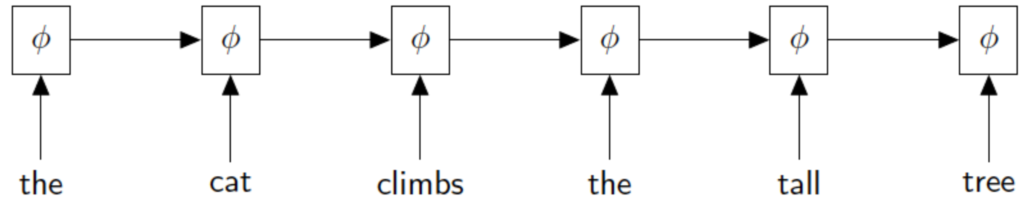
Meishan Zhang, Yue Zhang and Guohong Fu. End-to-End Neural Relation Extraction with Global Optimization. EMNLP 2017.

How to Use Tree or Graph Structures?

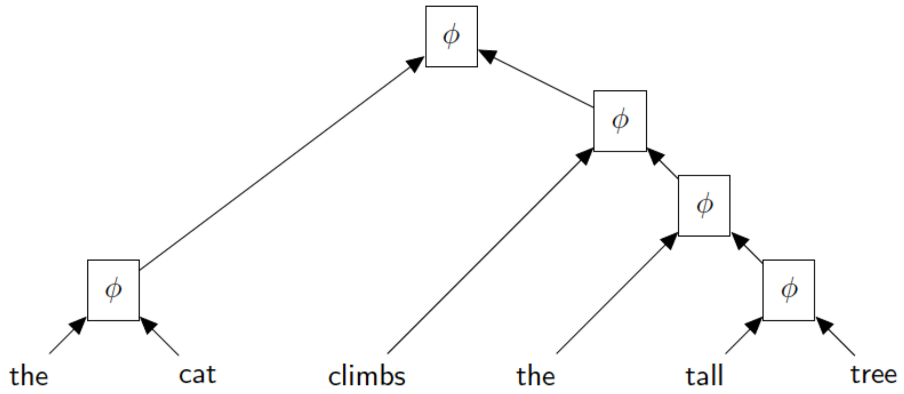
- As Information Extraction Rules
- As Input Features
- As Input Structures
- As Structured Prediction

Recurrent vs. Recursive Neural Networks

- Recurrent Neural Networks
  - Composing sequentially



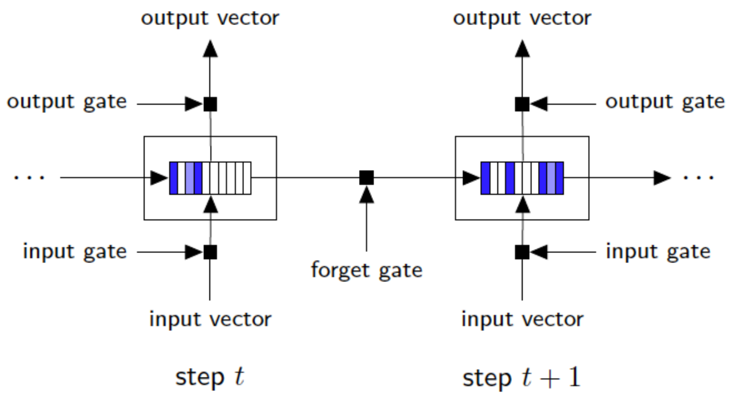
- Recursive Neural Networks
  - Use parse trees as input structures
  - Composing according to parsing structures



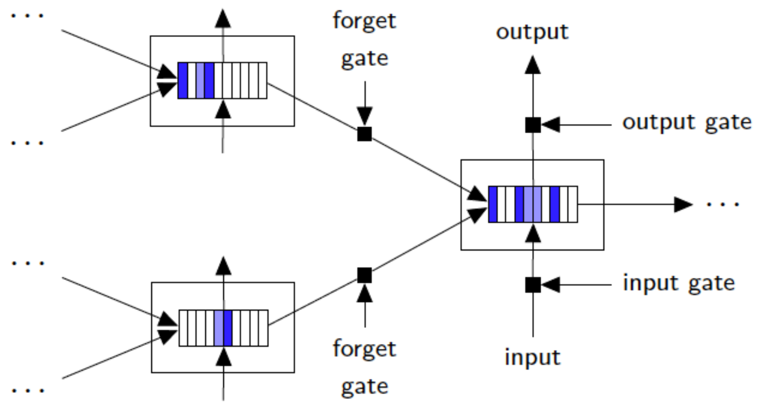
Richard Socher, Cliff Chiung-Yu Lin, Andrew Y. Ng and Christopher D. Manning. Parsing Natural Scenes And Natural Language With Recursive Neural Networks. ICML 2011.

Tree-LSTMs

Standard LSTM

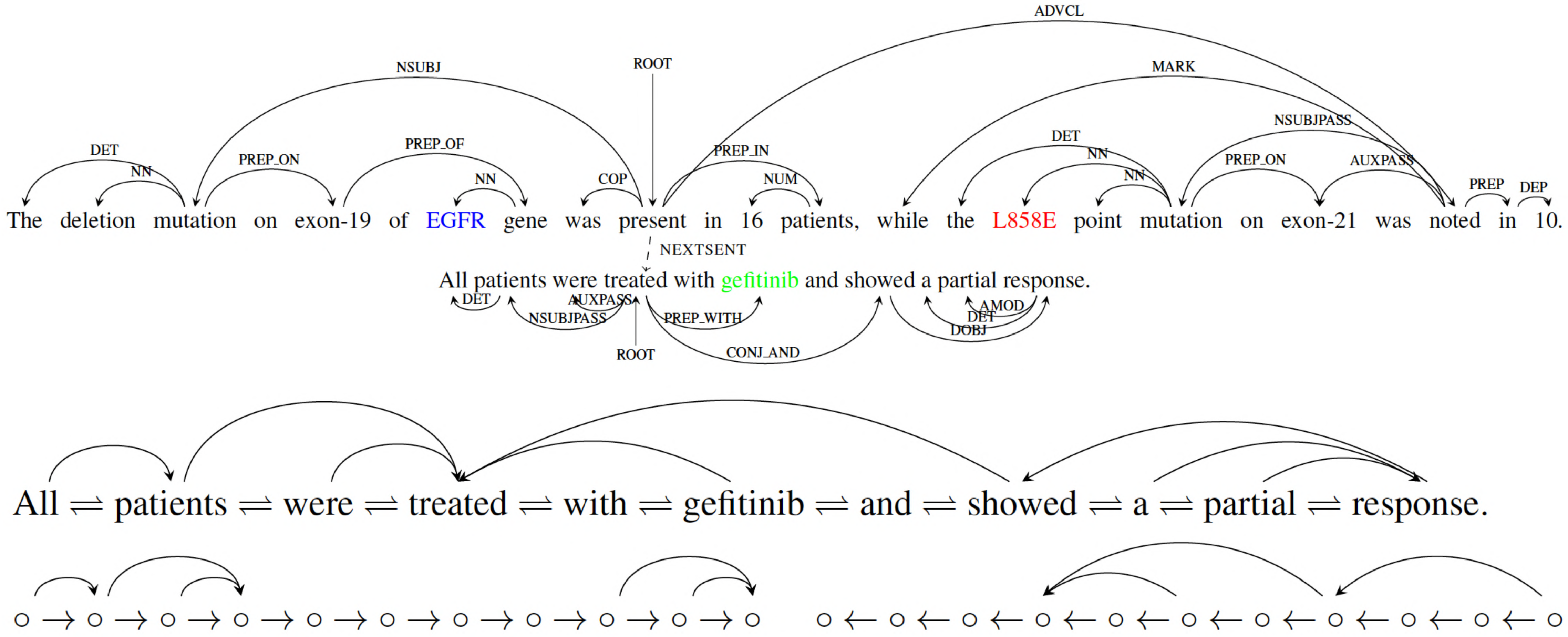


- Tree-LSTM



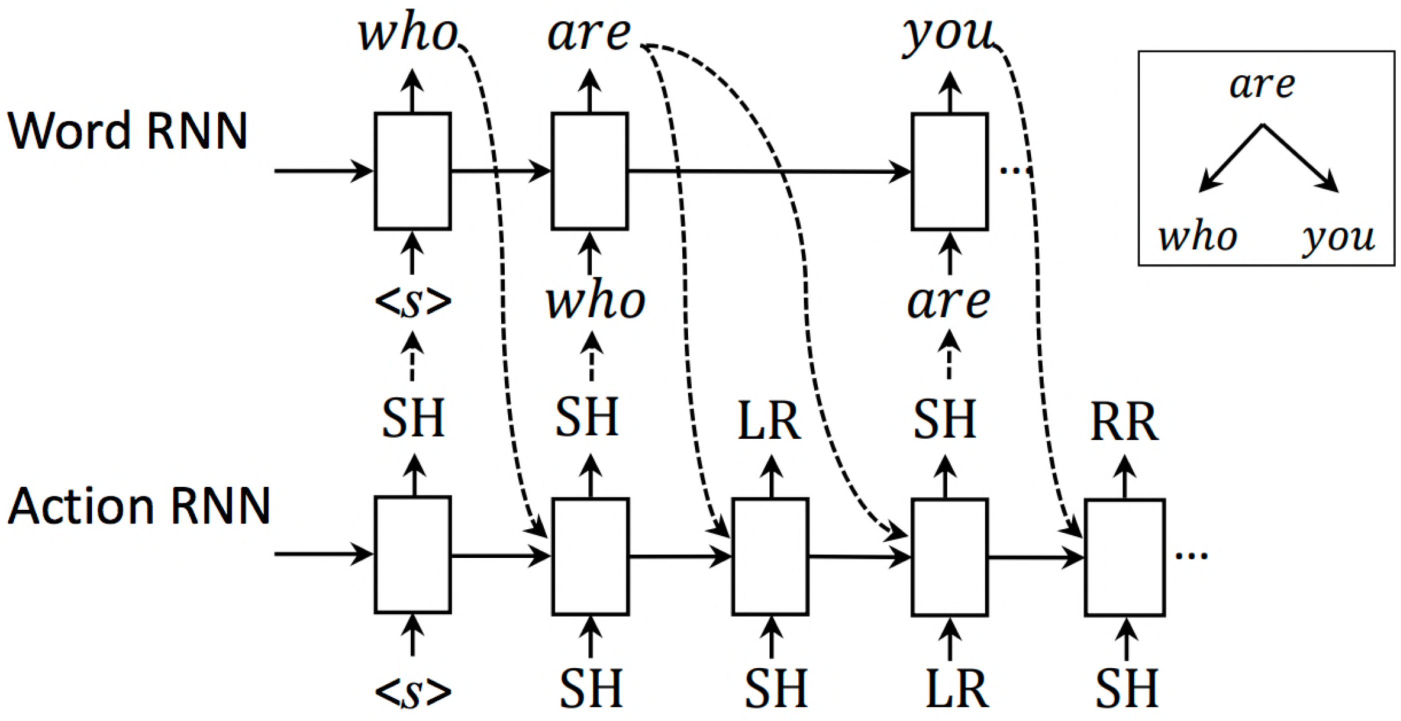
- Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. ACL 2015.
- Xiaodan Zhu, Parinaz Sobihani, and Hongyu Guo. 2015. Long short-term memory over recursive structures. ICML 2015.

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

Disfluency Detection

- Disfluency detection for speech recognition

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

- 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]

Outline

1. Syntactic and Semantic Parsing
2. Pseudo Data for Parsing
3. Applications of Parsing
4. Summary

Summary

- Syntactic and semantic parsing is one of the core task of NLP
- Recent advances
  - Grammar: universal dependency, semantic dependency graph
  - Data: large (pseudo) labeled data (multi-lingual/task, heterogeneous)
  - Algorithm: deep learning for semantic dependency graph parsing
- More and more applications
  - As Information Extraction Rules
  - As Input Features
  - As Input Structures
  - As Structured Prediction

Thanks!

http://ir.hit.edu.cn/~car/