Natural Language Processing

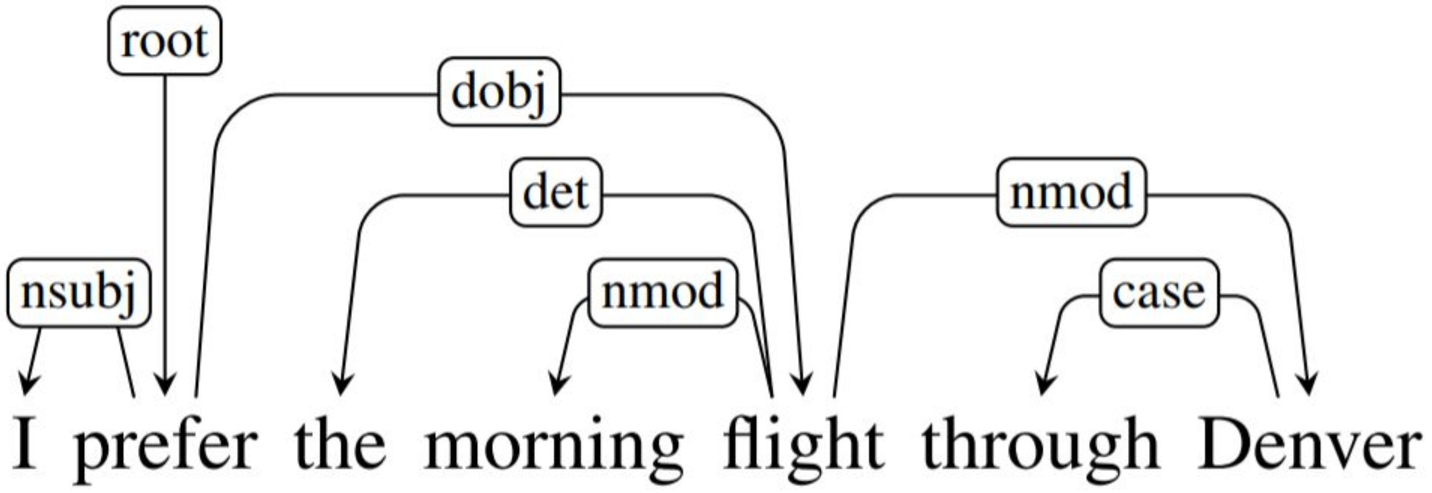
Syntactic parsing

Yulia Tsvetkov

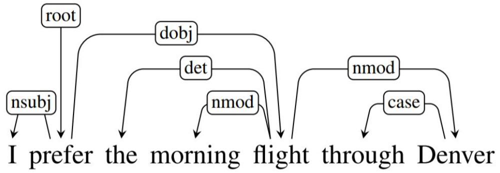
yuliats@cs.washington.edu

PAUL G. ALLEN SCHOOL OF COMPUTER SCIENCE & ENGINEERING

Dependency representation



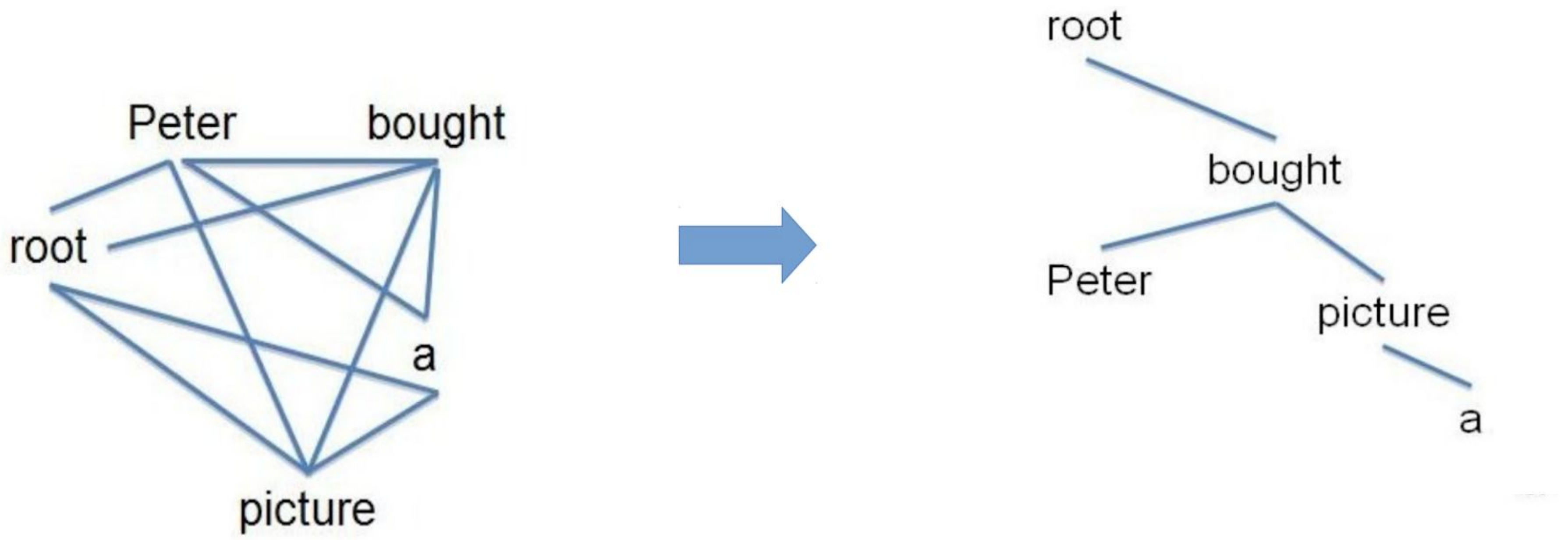
Dependency representation



- A dependency structure can be defined as a directed graph G, consisting of
  - a set V of nodes – vertices, *words, punctuation, morphemes*
  - a set A of arcs – directed edges,
  - a linear precedence order < on V (word order).
- Labeled graphs
  - nodes in V are labeled with word forms (and annotation).
  - arcs in A are labeled with dependency types
  - $L = \{l_1, \ldots, l_{|L|}\}$ is the set of permissible arc labels;
  - Every arc in A is a triple (i,j,k), representing a dependency from w_i to w_j with label l_k.

Parsing problem

- This is equivalent to finding a spanning tree in the complete graph containing all possible arcs

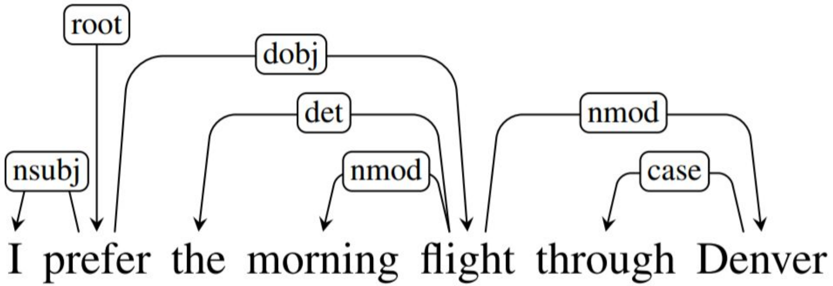


Parsing algorithms

- Transition based
  - greedy choice of local transitions guided by a good classifier
  - deterministic
  - MaltParser (Nivre et al. 2008)
- Graph based
  - Minimum Spanning Tree for a sentence
  - McDonald et al.'s (2005) MSTParser
  - Martins et al.'s (2009) Turbo Parser

Transition Based Parsing

- greedy discriminative dependency parser
- motivated by a stack-based approach called **shift-reduce parsing** originally developed for analyzing programming languages (Aho & Ullman, 1972).
- Nivre 2003



Configuration

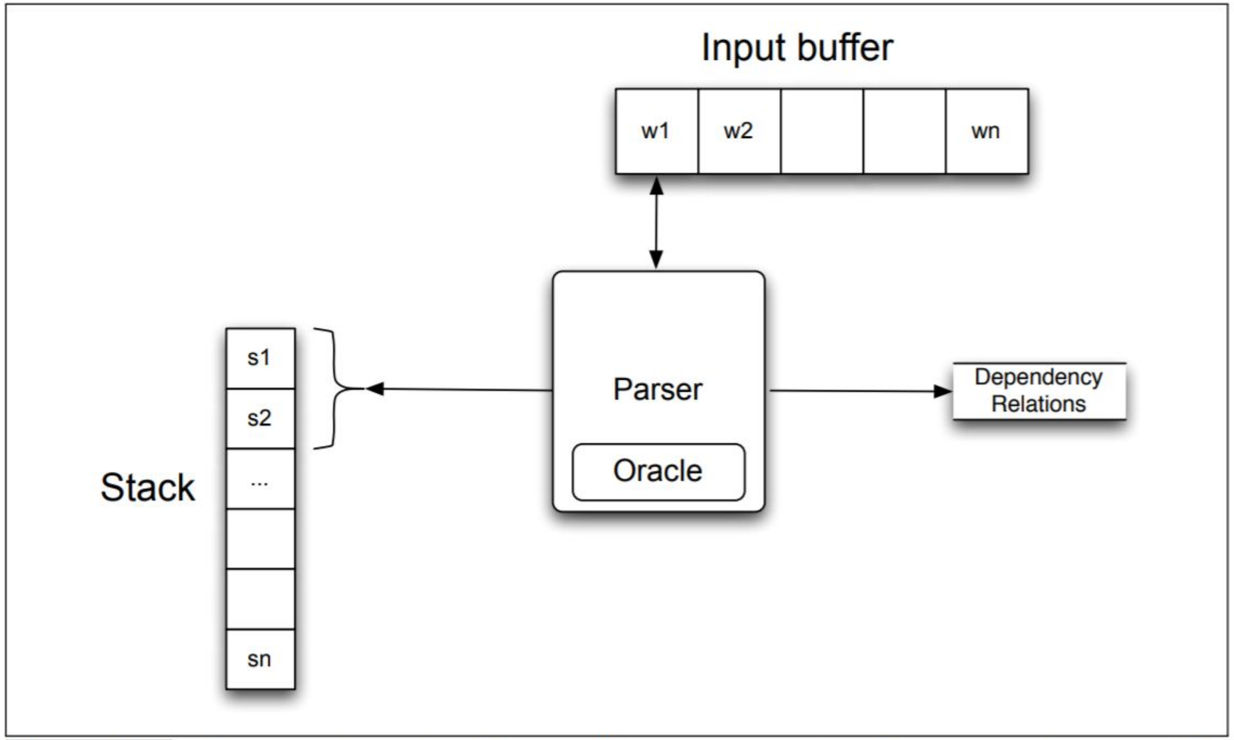


Figure 13.5 Basic transition-based parser. The parser examines the top two elements of the stack and selects an action based on consulting an oracle that examines the current configuration.

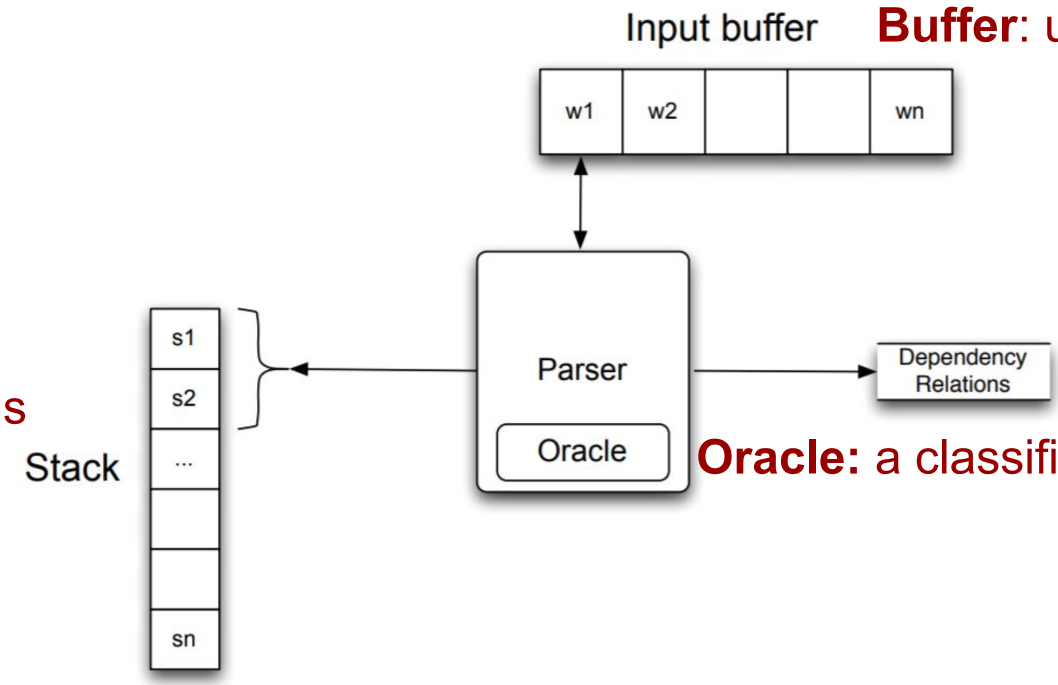
$$C = (\sigma, \beta, A)$$

Configuration

Buffer: unprocessed words

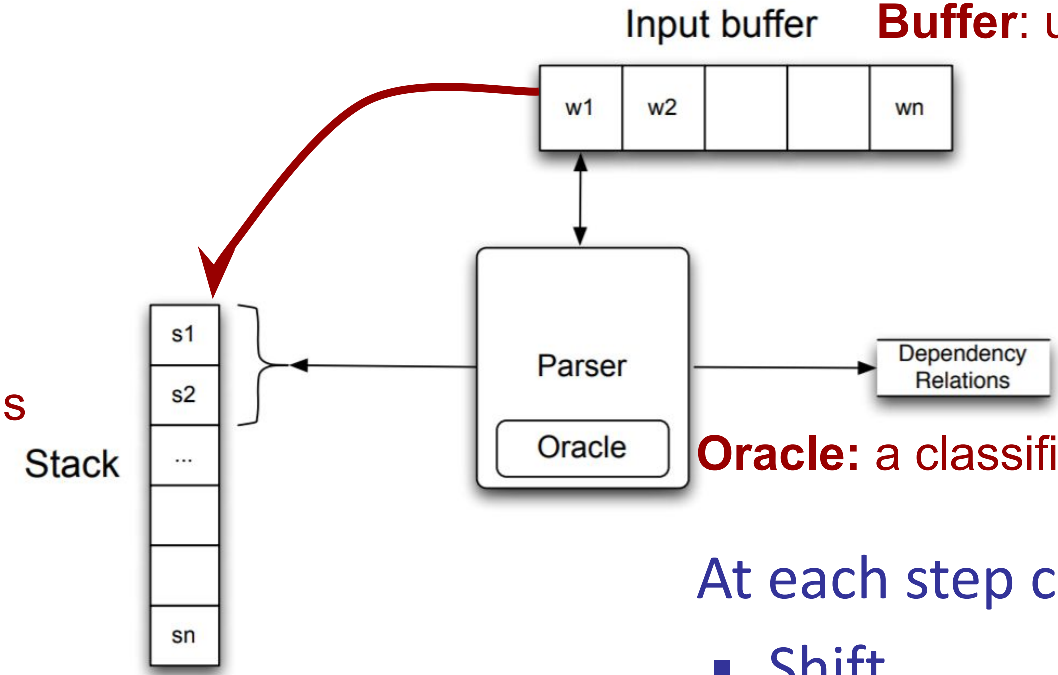
Stack: partially processed words

Oracle: a classifier



$$C_{\text{initial}} = ([\text{ROOT}], \boldsymbol{w}, \varnothing)$$

Operations



Buffer: unprocessed words

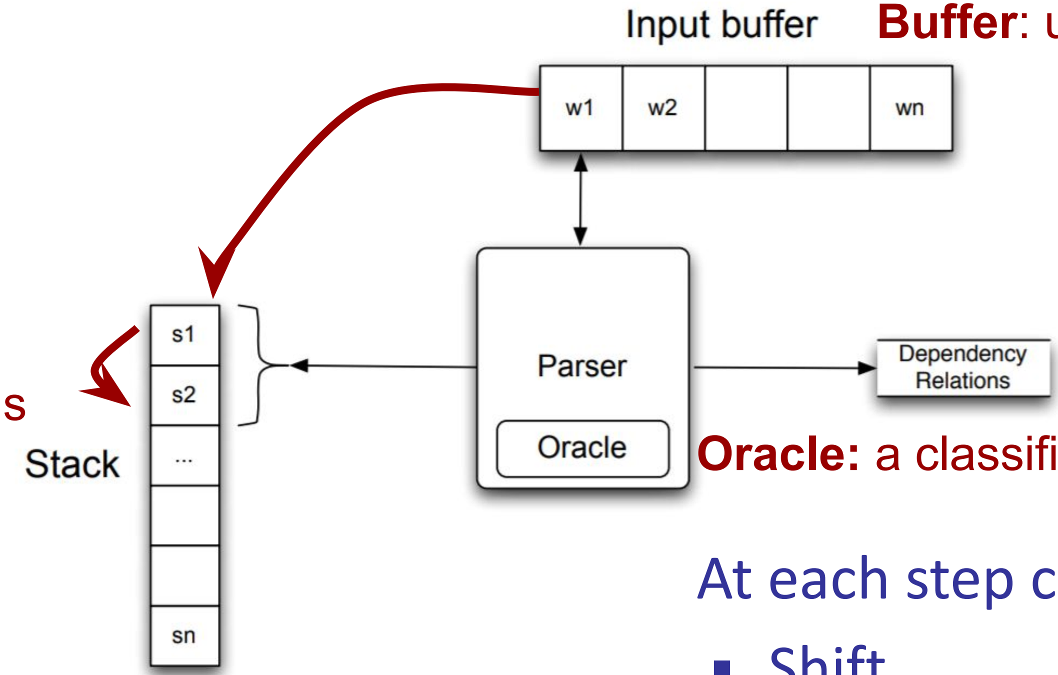
Stack: partially processed words

Oracle: a classifier

At each step choose:

- Shift

Operations



Buffer: unprocessed words

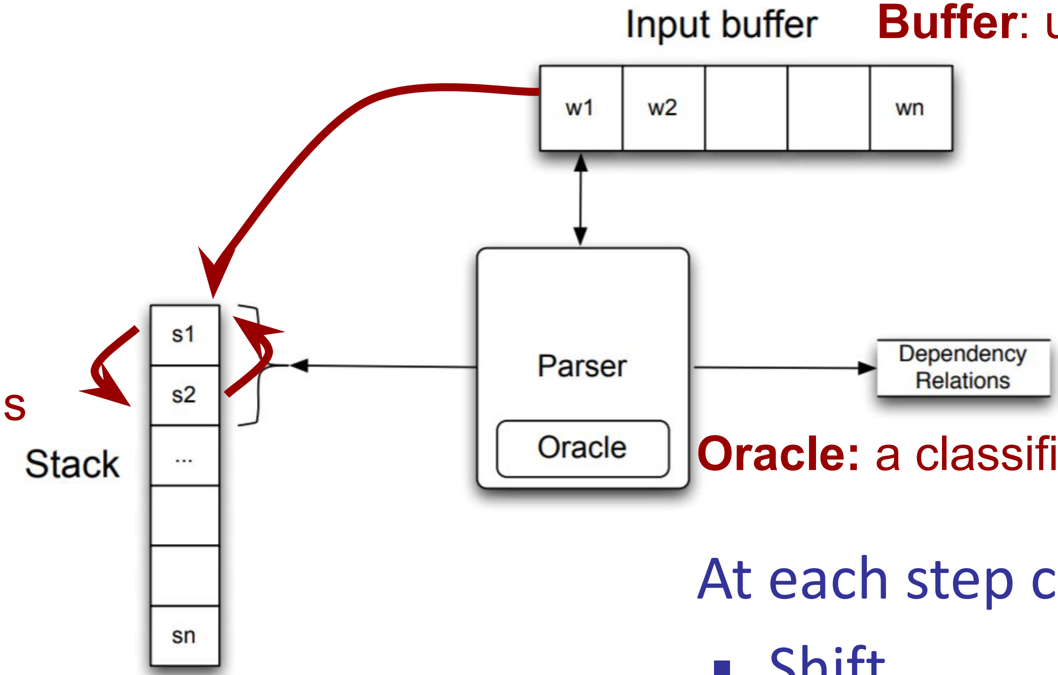
Stack: partially processed words

Oracle: a classifier

At each step choose:

- Shift
- Reduce left

Operations



Buffer: unprocessed words

Stack: partially processed words

Oracle: a classifier

$$C_{\text{accept}} = ([\text{ROOT}], \varnothing, A)$$

At each step choose:

- Shift
- LeftArc or Reduce left
- RightArc or Reduce right

Shift-Reduce Parsing

Configuration:

- Stack, Buffer, Oracle, Set of dependency relations

Operations by a classifier at each step:

- Shift
  - remove w1 from the buffer, add it to the top of the stack as s1
- LeftArc or Reduce left
  - assert a head-dependent relation between s1 and s2 ($s1 \longrightarrow s2$)
  - remove s2 from the stack
- RightArc or Reduce right
  - assert a head-dependent relation between s2 and s1 ($s2 \longrightarrow s1$)
  - remove s1 from the stack

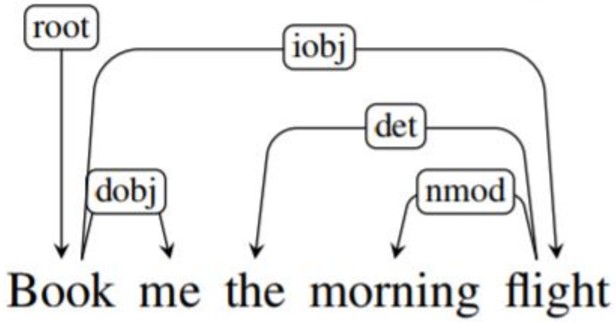
Shift-Reduce Parsing (Arc-standard)

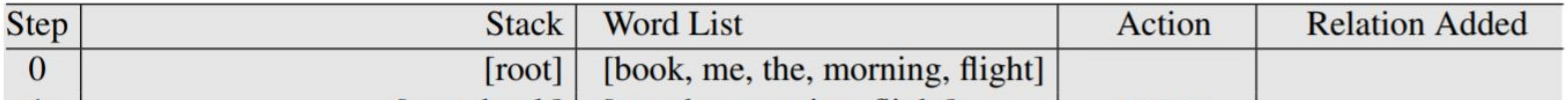
$$C_{\text{initial}} = ([\text{ROOT}], \boldsymbol{w}, \varnothing)$$

Book me the morning flight

Step	Stack	Word List	Action	Relation Added

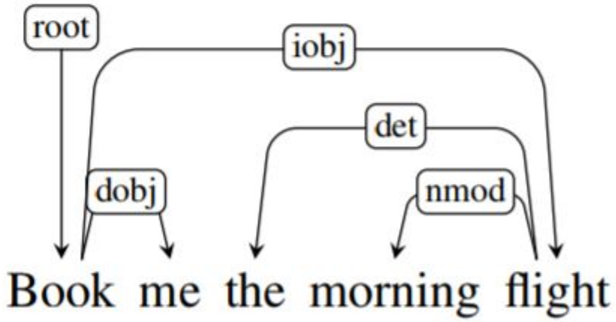
Shift-Reduce Parsing





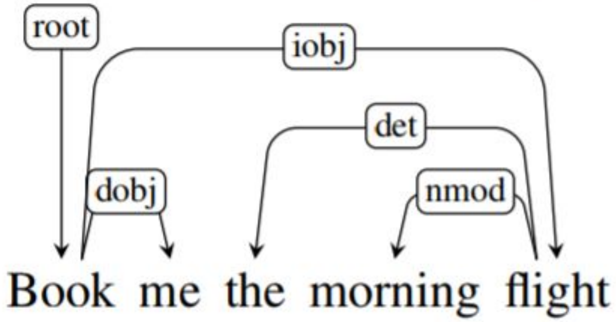
Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]		

Shift-Reduce Parsing



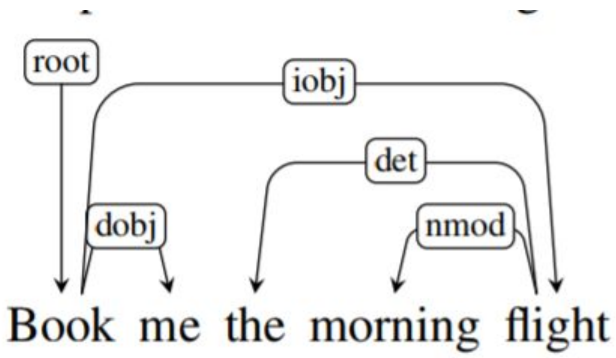
Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]		

Shift-Reduce Parsing



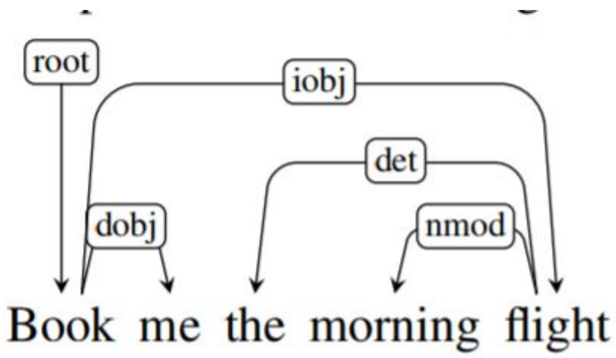
Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]		

Shift-Reduce Parsing



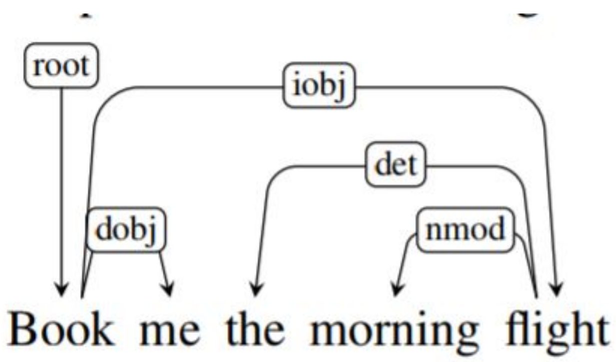
Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	

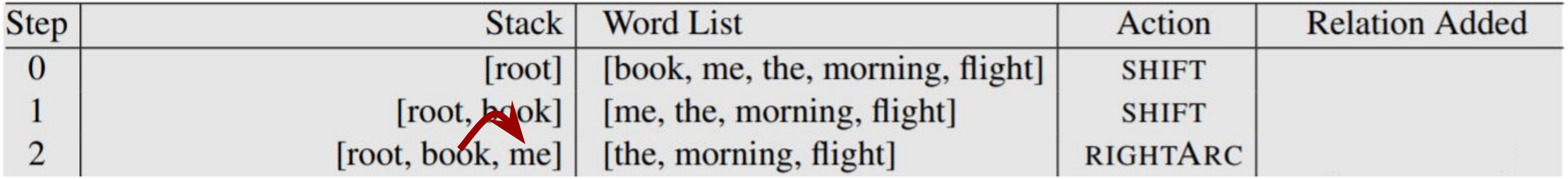
Shift-Reduce Parsing



Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]		

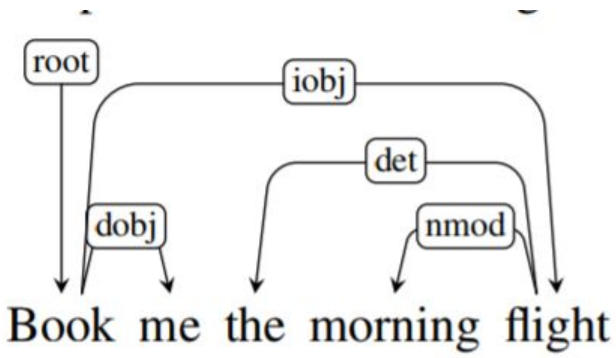
Shift-Reduce Parsing





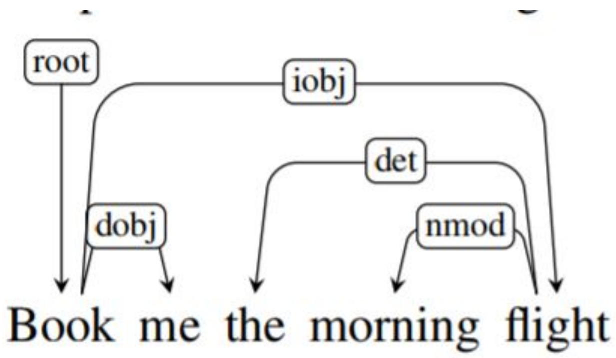
Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	

Shift-Reduce Parsing



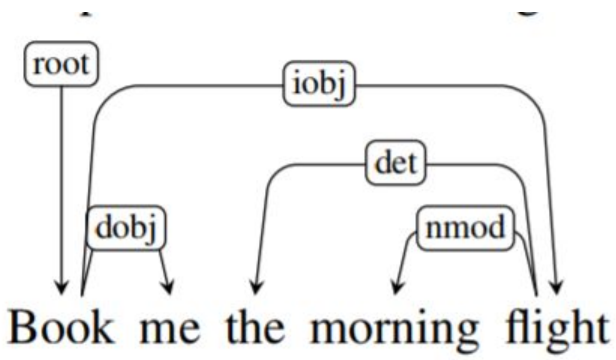
Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book → me)

Shift-Reduce Parsing



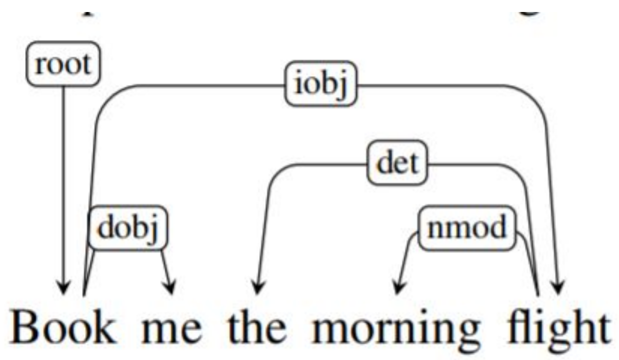
Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book $\rightarrow$ me)
3	[root, book]	[the, morning, flight]		

Shift-Reduce Parsing



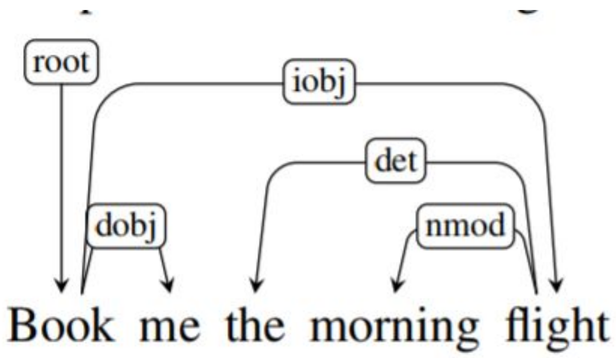
Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book → me)
3	[root, book]	[the, morning, flight]	SHIFT	

Shift-Reduce Parsing



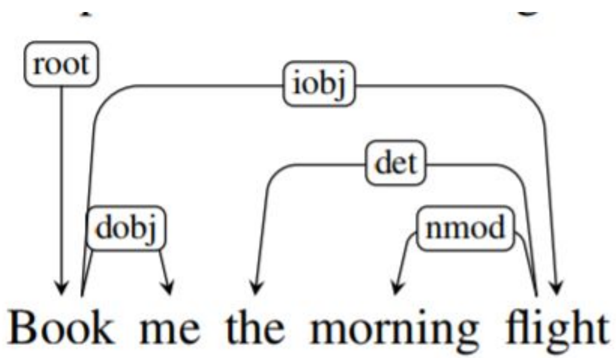
Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book → me)
3	[root, book]	[the, morning, flight]	SHIFT	
4	[root, book, the]	[morning, flight]		

Shift-Reduce Parsing



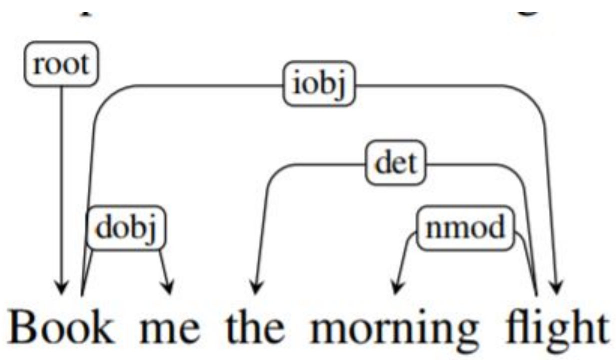
Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book $\rightarrow$ me)
3	[root, book]	[the, morning, flight]	SHIFT	
4	[root, book, the]	[morning, flight]	SHIFT	

Shift-Reduce Parsing



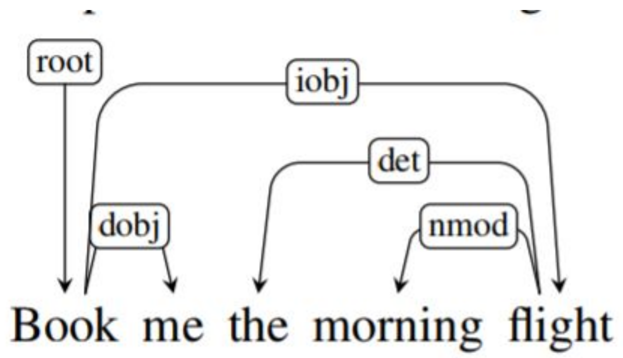
Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book → me)
3	[root, book]	[the, morning, flight]	SHIFT	
4	[root, book, the]	[morning, flight]	SHIFT	
5	[root, book, the, morning]	[flight]		

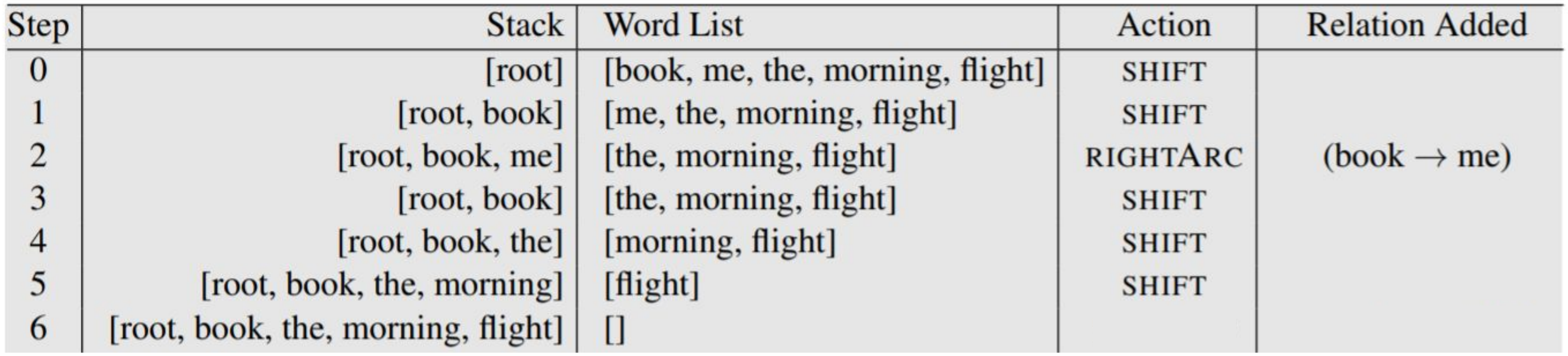
Shift-Reduce Parsing



Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book → me)
3	[root, book]	[the, morning, flight]	SHIFT	
4	[root, book, the]	[morning, flight]	SHIFT	
5	[root, book, the, morning]	[flight]	SHIFT	

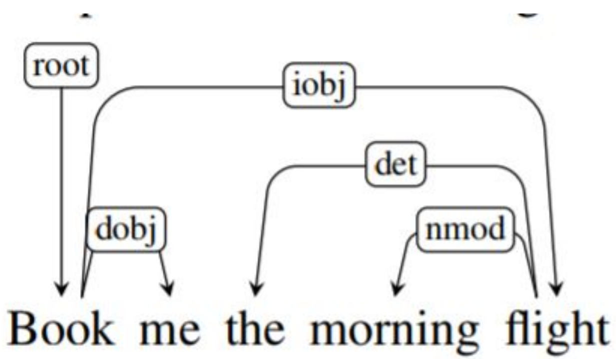
Shift-Reduce Parsing





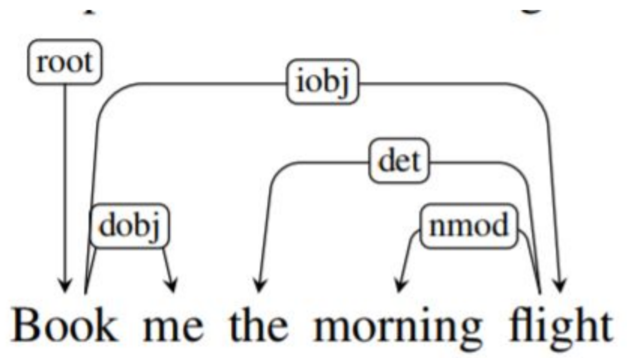
Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book → me)
3	[root, book]	[the, morning, flight]	SHIFT	
4	[root, book, the]	[morning, flight]	SHIFT	
5	[root, book, the, morning]	[flight]	SHIFT	
6	[root, book, the, morning, flight]	[]		

Shift-Reduce Parsing



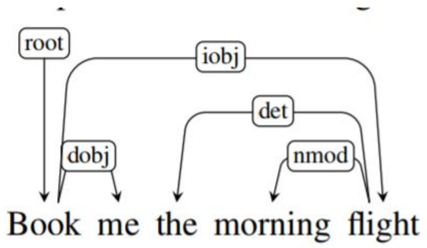
Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book → me)
3	[root, book]	[the, morning, flight]	SHIFT	
4	[root, book, the]	[morning, flight]	SHIFT	
5	[root, book, the, morning]	[flight]	SHIFT	
6	[root, book, the, morning, flight]	[]	LEFTARC	(morning ← flight)

Shift-Reduce Parsing



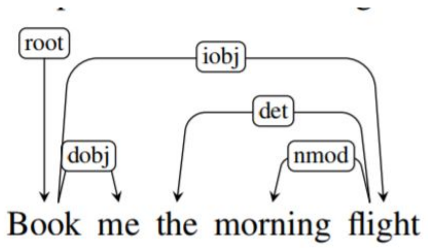
Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book → me)
3	[root, book]	[the, morning, flight]	SHIFT	
4	[root, book, the]	[morning, flight]	SHIFT	
5	[root, book, the, morning]	[flight]	SHIFT	
6	[root, book, the, morning, flight]	[]	LEFTARC	(morning ← flight)
7	[root, book, the, flight]	[]	LEFTARC	(the ← flight)

Shift-Reduce Parsing



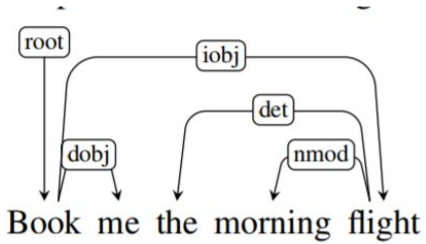
Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book → me)
3	[root, book]	[the, morning, flight]	SHIFT	
4	[root, book, the]	[morning, flight]	SHIFT	
5	[root, book, the, morning]	[flight]	SHIFT	
6	[root, book, the, morning, flight]	[]	LEFTARC	(morning ← flight)
7	[root, book, the, flight]	[]	LEFTARC	(the ← flight)
8	[root, book, flight]	[]	RIGHTARC	(book → flight)

Shift-Reduce Parsing



Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book → me)
3	[root, book]	[the, morning, flight]	SHIFT	
4	[root, book, the]	[morning, flight]	SHIFT	
5	[root, book, the, morning]	[flight]	SHIFT	
6	[root, book, the, morning, flight]	[]	LEFTARC	(morning ← flight)
7	[root, book, the, flight]	[]	LEFTARC	(the ← flight)
8	[root, book, flight]	[]	RIGHTARC	(book → flight)
9	[root, book]	[]	RIGHTARC	(root → book)

Shift-Reduce Parsing



$$C_{\text{accept}} = ([\textsc{Root}], \varnothing, A)$$

Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book → me)
3	[root, book]	[the, morning, flight]	SHIFT	
4	[root, book, the]	[morning, flight]	SHIFT	
5	[root, book, the, morning]	[flight]	SHIFT	
6	[root, book, the, morning, flight]	[]	LEFTARC	(morning ← flight)
7	[root, book, the, flight]	[]	LEFTARC	(the ← flight)
8	[root, book, flight]	[]	RIGHTARC	(book → flight)
9	[root, book]	[]	RIGHTARC	(root → book)
10	[root]	[]	Done	

Shift-Reduce Parsing

Configuration:

- Stack, Buffer, Oracle, Set of dependency relations

Operations by a classifier at each step:

Complexity?

- Shift
  - remove w1 from the buffer, add it to the top of the stack as s1
- LeftArc or Reduce left
  - assert a head-dependent relation between
  - remove s2 from the stack
- RightArc or Reduce right
  - assert a head-dependent relation between s2 and s1
  - remove s1 from the stack

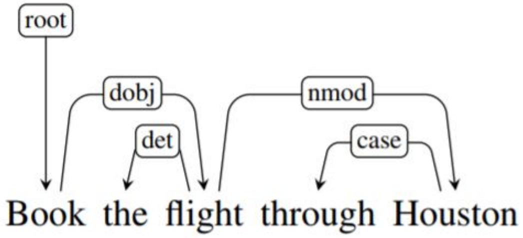
Oracle decisions can correspond to unlabeled or labeled arcs

Training an Oracle

- Oracle is a supervised classifier that learns a function from the configuration to the next operation
- How to extract the training set?

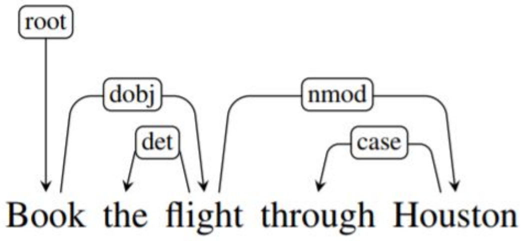
Training an Oracle

- How to extract the training set?
  - if LeftArc ⟶ LeftArc
  - if RightArc
    - if s1 dependents have been processed ⟶ RightArc
  - else ⟶ Shift



Training an Oracle

- How to extract the training set?
  - if LeftArc ⟶ LeftArc
  - if RightArc
    - if s1 dependents have been processed ⟶ RightArc
  - else ⟶ Shift



Step	Stack	Word List	Predicted Action
0	[root]	[book, the, flight, through, houston]	SHIFT
1	[root, book]	[the, flight, through, houston]	SHIFT
2	[root, book, the]	[flight, through, houston]	SHIFT
3	[root, book, the, flight]	[through, houston]	LEFTARC
4	[root, book, flight]	[through, houston]	SHIFT
5	[root, book, flight, through]	[houston]	SHIFT
6	[root, book, flight, through, houston]	[]	LEFTARC
7	[root, book, flight, houston]	[]	RIGHTARC
8	[root, book, flight]	[]	RIGHTARC
9	[root, book]	[]	RIGHTARC

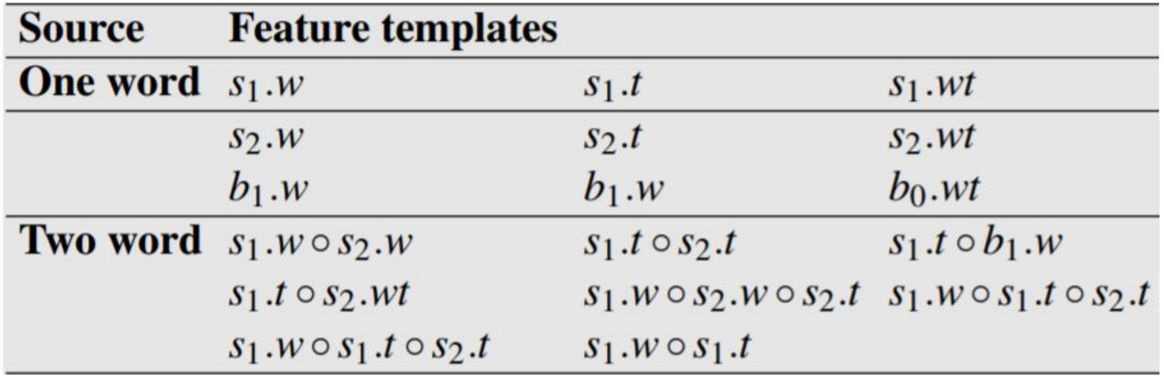
Training an Oracle

- Oracle is a supervised classifier that learns a function from the configuration to the next operation
- How to extract the training set?
  - if LeftArc $\longrightarrow$ LeftArc
  - if RightArc
    - if s1 dependents have been processed $\longrightarrow$ RightArc
  - else $\longrightarrow$ Shift
- What features to use?

Features

- POS, word-forms, lemmas on the stack/buffer
- morphological features for some languages
- previous relations
- conjunction features (e.g. Zhang&Clark'08; Huang&Sagae'10; Zhang&Nivre'11)

$$\langle s_1.w = \mathit{flights}, op = \mathit{shift}\rangle$$
$$\langle s_2.w = \mathit{canceled}, op = \mathit{shift}\rangle$$
$$\langle s_1.t = \mathit{NNS}, op = \mathit{shift}\rangle$$
$$\langle s_2.t = \mathit{VBD}, op = \mathit{shift}\rangle$$
$$\langle b_1.w = \mathit{to}, op = \mathit{shift}\rangle$$
$$\langle b_1.t = \mathit{TO}, op = \mathit{shift}\rangle$$
$$\langle s_1.wt = \mathit{flightsNNS}, op = \mathit{shift}\rangle$$
$$\langle s_1.t \circ s_2.t = \mathit{NNSVBD}, op = \mathit{shift}\rangle$$

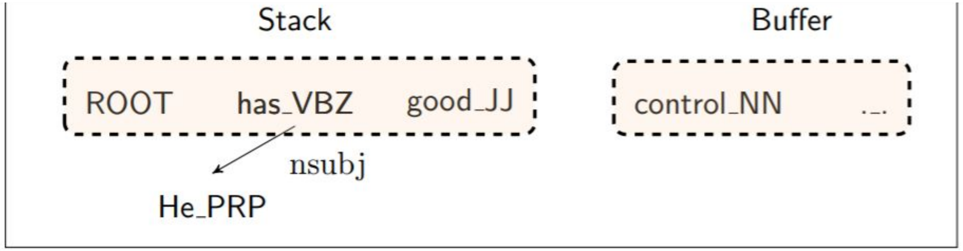


Source	Feature templates		
One word	$s_1.w$	$s_1.t$	$s_1.wt$
	$s_2.w$	$s_2.t$	$s_2.wt$
	$b_1.w$	$b_1.w$	$b_0.wt$
Two word	$s_1.w \circ s_2.w$	$s_1.t \circ s_2.t$	$s_1.t \circ b_1.w$
	$s_1.t \circ s_2.wt$	$s_1.w \circ s_2.w \circ s_2.t$	$s_1.w \circ s_1.t \circ s_2.t$
	$s_1.w \circ s_1.t \circ s_2.t$	$s_1.w \circ s_1.t$	

Learning

- Before 2014: SVMs,
- After 2014: Neural Nets

Chen & Manning 2014



binary, sparse
dim $=10^6 \sim 10^7$

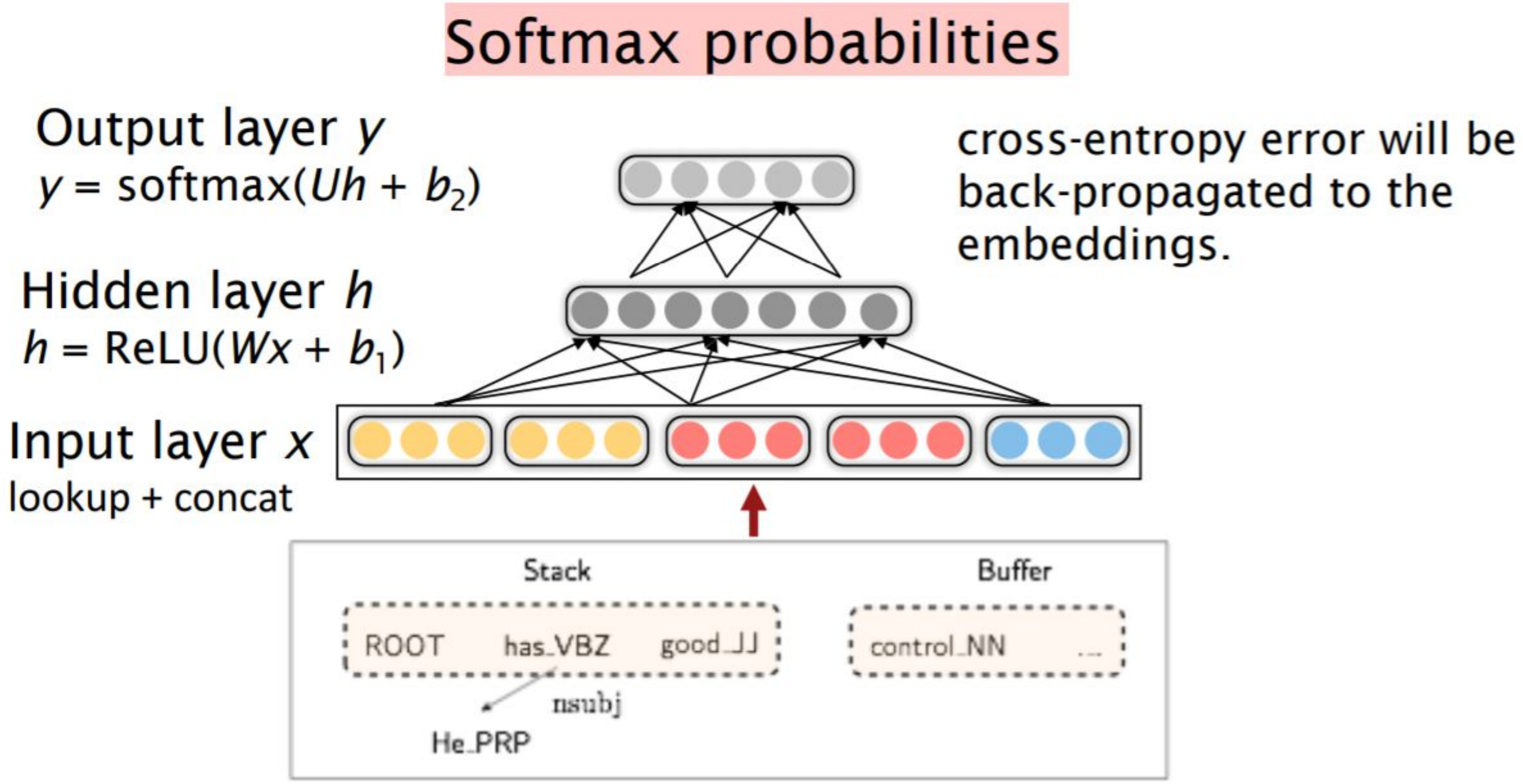
0	0	0	1	0	0	1	0	…	0	0	1	0

Indicator features

$$s_2.w = \text{has} \wedge s_2.t = \text{VBZ}$$
$$s_1.w = \text{good} \wedge s_1.t = \text{JJ} \wedge b_1.w = \text{control}$$
$$lc(s_2).t = \text{PRP} \wedge s_2.t = \text{VBZ} \wedge s_1.t = \text{JJ}$$
$$lc(s_2).w = \text{He} \wedge lc(s_2).l = \text{nsubj} \wedge s_2.w = \text{has}$$

Slides by Danqi Chen & Chris Manning

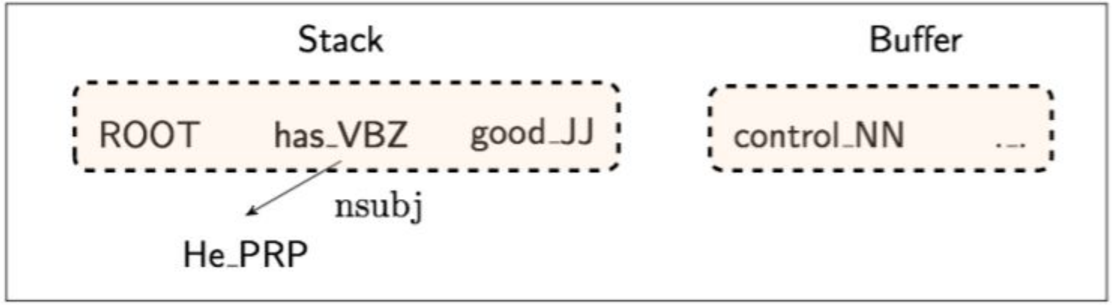
Chen & Manning 2014

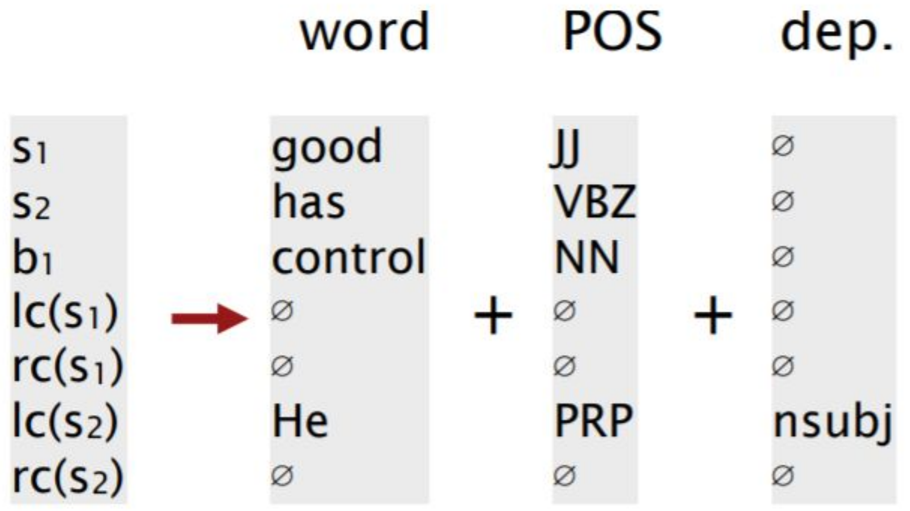


cross-entropy error will be back-propagated to the embeddings.

Chen & Manning 2014

- Features
  - s1, s2, s3, b1, b2, b3
  - leftmost/rightmost children of s1 and s2
  - leftmost/rightmost grandchildren of s1 and s2
  - POS tags for the above
  - arc labels for children/grandchildren

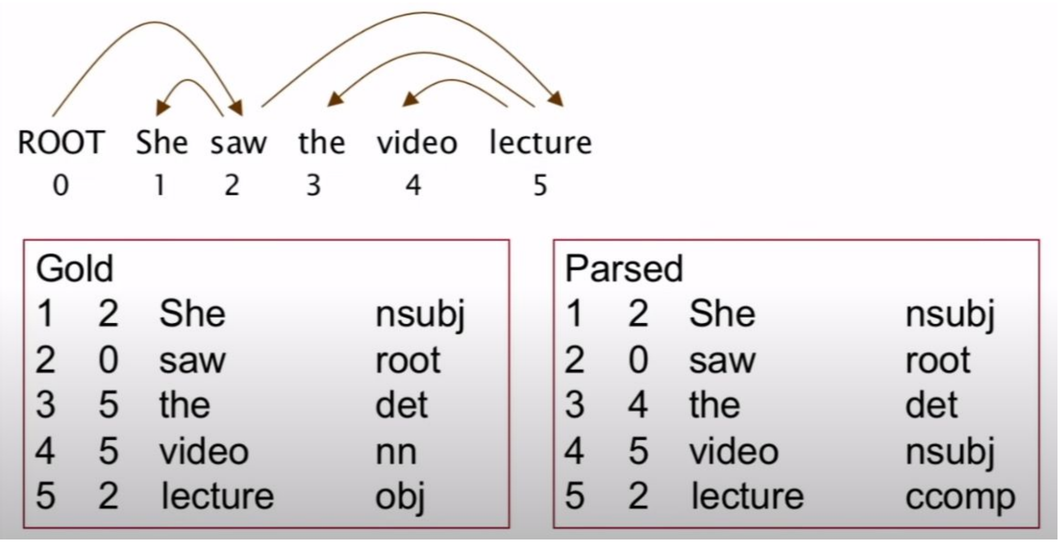




	word		POS		dep.
s_1	good		JJ		∅
s_2	has		VBZ		∅
b_1	control		NN		∅
$lc(s_1)$	∅	+	∅	+	∅
$rc(s_1)$	∅		∅		∅
$lc(s_2)$	He		PRP		nsubj
$rc(s_2)$	∅		∅		∅

Evaluation of Dependency Parsers

$$\frac{\#correct\ dependencies}{\#of\ dependencies}$$



Gold

1	2	She	nsubj
2	0	saw	root
3	5	the	det
4	5	video	nn
5	2	lecture	obj

Parsed

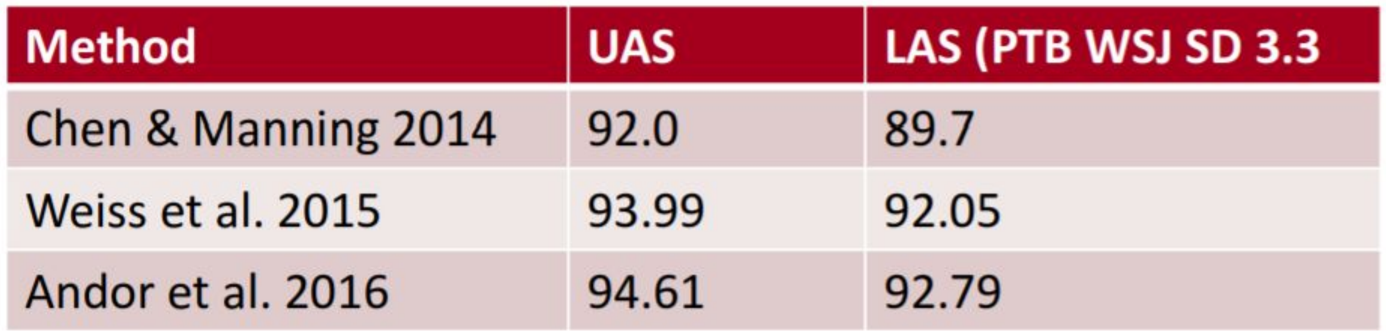
1	2	She	nsubj
2	0	saw	root
3	4	the	det
4	5	video	nsubj
5	2	lecture	ccomp

- LAS - labeled attachment score
- UAS - unlabeled attachment score

Chen & Manning 2014

Parser	UAS	LAS	sent. / s
MaltParser	89.8	87.2	469
MSTParser	91.4	88.1	10
TurboParser	**92.3***	89.6*	8
C & M 2014	92.0	**89.7**	**654**

Follow-up

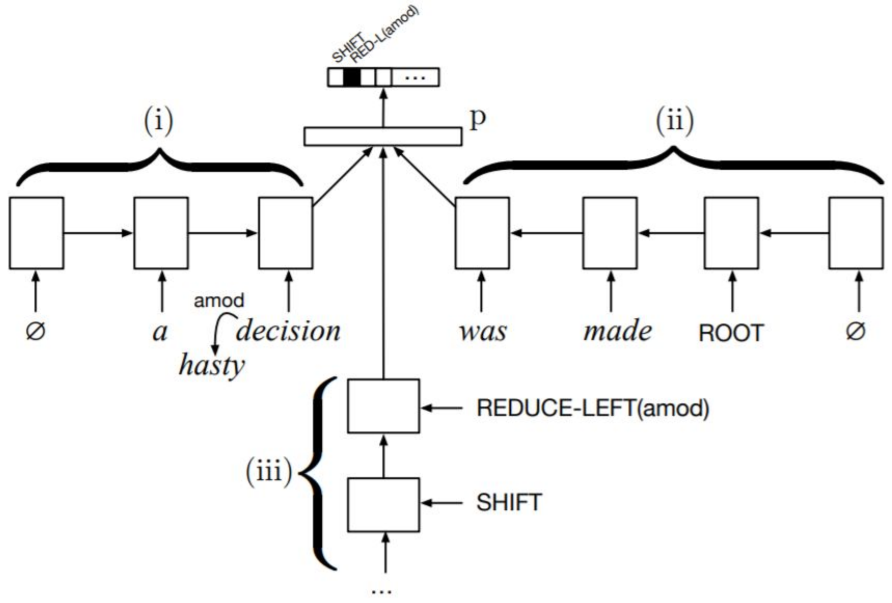


Method	UAS	LAS (PTB WSJ SD 3.3
Chen & Manning 2014	92.0	89.7
Weiss et al. 2015	93.99	92.05
Andor et al. 2016	94.61	92.79

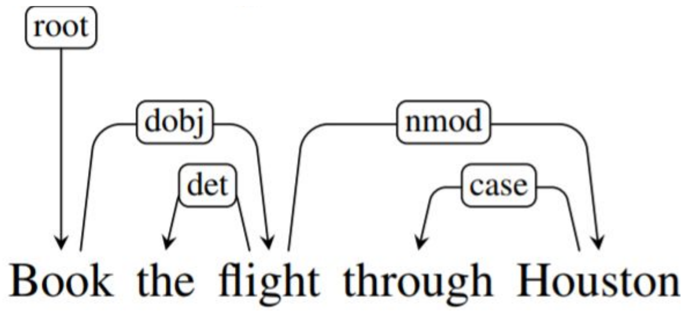
Stack LSTMs (Dyer et al. 2015)

Transition-Based Dependency Parsing with Stack Long Short-Term Memory

Chris Dyer♣♠ **Miguel Ballesteros**♢♠ **Wang Ling**♠ **Austin Matthews**♠ **Noah A. Smith**♠
♣Marianas Labs ♢NLP Group, Pompeu Fabra University ♠Carnegie Mellon University
chris@marianaslabs.com, miguel.ballesteros@upf.edu,
{lingwang,austinma,nasmith}@cs.cmu.edu

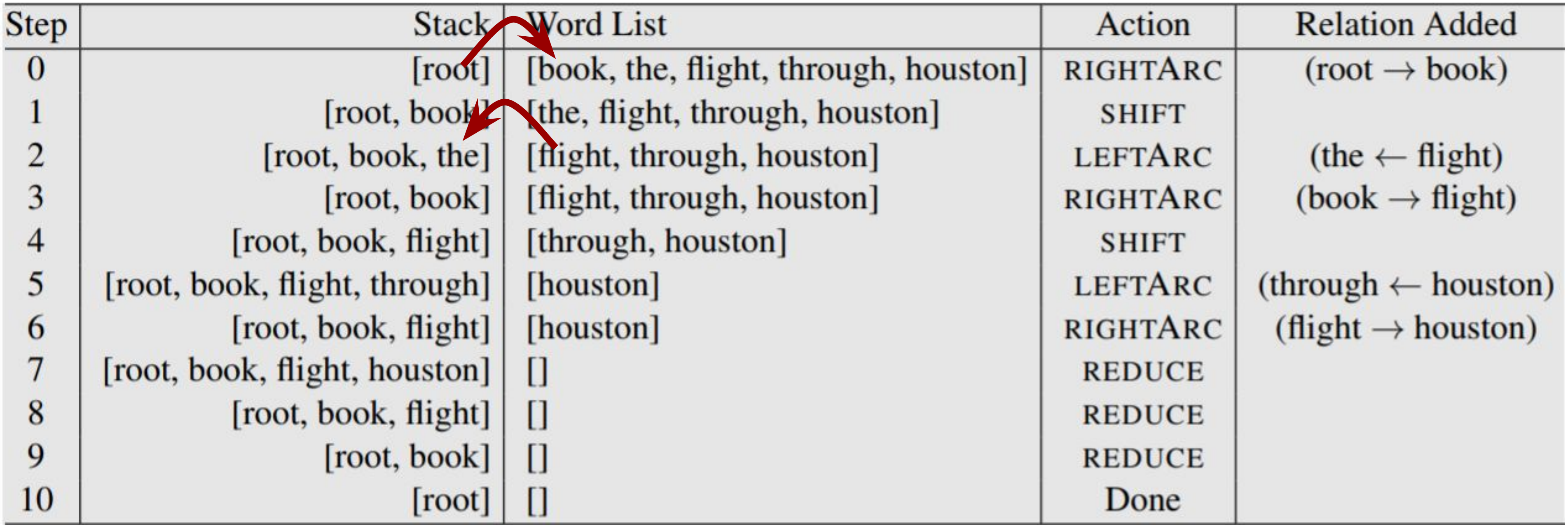


Arc-Eager version



- LEFTARC: Assert a head-dependent relation between s1 and b1; pop the stack.
- RIGHTARC: Assert a head-dependent relation between s1 and b1; shift b1 to be s1.
- SHIFT: Remove b1 and push it to be s1.
- REDUCE: Pop the stack.

Arc-Eager



Step	Stack	Word List	Action	Relation Added
0	[root]	[book, the, flight, through, houston]	RIGHTARC	(root → book)
1	[root, book]	[the, flight, through, houston]	SHIFT	
2	[root, book, the]	[flight, through, houston]	LEFTARC	(the ← flight)
3	[root, book]	[flight, through, houston]	RIGHTARC	(book → flight)
4	[root, book, flight]	[through, houston]	SHIFT	
5	[root, book, flight, through]	[houston]	LEFTARC	(through ← houston)
6	[root, book, flight]	[houston]	RIGHTARC	(flight → houston)
7	[root, book, flight, houston]	[]	REDUCE	
8	[root, book, flight]	[]	REDUCE	
9	[root, book]	[]	REDUCE	
10	[root]	[]	Done	

Parsing algorithms

- Transition based
  - greedy choice of local transitions guided by a good classifier
  - deterministic
  - MaltParser (Nivre et al. 2008), Stack LSTM (Dyer et al. 2015)
- Graph based
  - Minimum Spanning Tree for a sentence
  - non-projective
  - globally optimized
  - McDonald et al.'s (2005) MSTParser
  - Martins et al.'s (2009) Turbo Parser

Summary

- Transition-based
  - + Fast
  - + Rich features of context
  - - Greedy decoding
- Graph-based
  - + Exact or close to exact decoding
  - - Weaker features

Well-engineered versions of the approaches achieve comparable accuracy (on English), but make different errors

$\rightarrow$ combining the strategies results in a substantial boost in performance