Greedy, Joint Syntactic-Semantic Parsing with Stack LSTMs

Swabha Swayamdipta[1] Miguel Ballesteros[2]
Chris Dyer[3] Noah A. Smith[4]

[1]School of Computer Science, Carnegie Mellon University
Pittsburgh, USA

[2]NLP Group, Universitat Pompeu Fabra
Barcelona, Spain

[3]Google DeepMind
London, UK

[4]Computer Science and Engineering, University of Washington
Seattle, USA

12th Aug
(CoNLL 2016)

Joint syntactic-semantic parsing

YM-style Syntactic dependency parsing + PropBank-style semantic role labeling

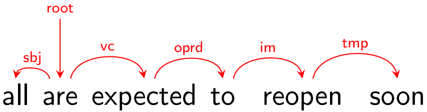
Joint syntactic-semantic parsing

YM-style Syntactic dependency parsing + PropBank-style semantic role labeling

all are expected to reopen soon

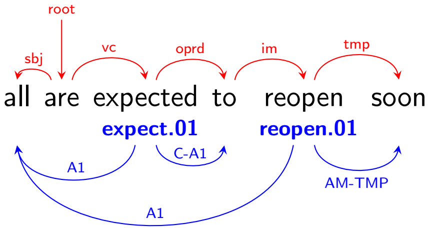
Joint syntactic-semantic parsing

YM-style Syntactic dependency parsing + PropBank-style semantic role labeling



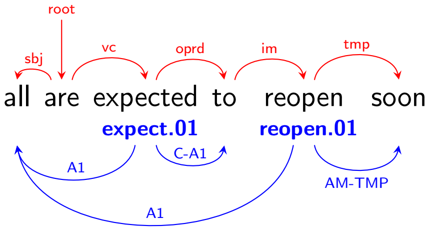
Joint syntactic-semantic parsing

YM-style Syntactic dependency parsing + PropBank-style semantic role labeling



Joint syntactic-semantic parsing

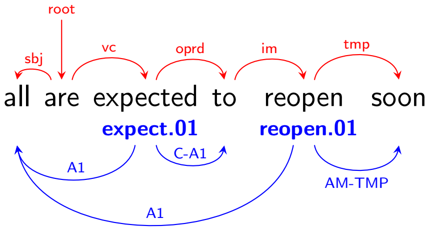
YM-style Syntactic dependency parsing + PropBank-style semantic role labeling



- Correspondence between syntactic and semantic dependencies [Levin and Hovav, 1996]

Joint syntactic-semantic parsing

YM-style Syntactic dependency parsing + PropBank-style semantic role labeling



- Correspondence between syntactic and semantic dependencies [Levin and Hovav, 1996]
- Language understanding: QA, relation extraction, text categorization

A little more about PropBank SRL

[Palmer et al., 2005]

A little more about PropBank SRL

[Palmer et al., 2005]

- Most common solution: pipeline syntax and semantics

A little more about PropBank SRL

[Palmer et al., 2005]

- Most common solution: pipeline syntax and semantics
- Pipelines involve *expensive* feature extraction step [Johansson, 2009, He et al., 2013]

A little more about PropBank SRL
[Palmer et al., 2005]

- Most common solution: pipeline syntax and semantics
- Pipelines involve *expensive* feature extraction step [Johansson, 2009, He et al., 2013]
- Our approach : incremental, joint parsing of syntax and semantics

A little more about PropBank SRL
[Palmer et al., 2005]

- Most common solution: pipeline syntax and semantics
- Pipelines involve *expensive* feature extraction step [Johansson, 2009, He et al., 2013]
- Our approach : incremental, joint parsing of syntax and semantics

Pipelines

- Have access to complete syntactic information

Incremental, joint approach

- No such access

A little more about PropBank SRL
[Palmer et al., 2005]

- Most common solution: pipeline syntax and semantics
- Pipelines involve *expensive* feature extraction step [Johansson, 2009, He et al., 2013]
- Our approach : incremental, joint parsing of syntax and semantics

Pipelines

- Have access to complete syntactic information
- Slow feature extraction step

Incremental, joint approach

- No such access
- Fast

Outline

Incremental algorithm

- Parse structure → sequence of transitions

Incremental algorithm

- Parse structure $\rightarrow$ sequence of transitions
- Transition : **shift** and **reduce** actions

Incremental algorithm

- Parse structure $\rightarrow$ sequence of transitions
- Transition : **shift** and **reduce** actions
- Data structures : *stack* and *buffer*

Incremental algorithm

- Parse structure $\rightarrow$ sequence of transitions
- Transition : **shift** and **reduce** actions
- Data structures : *stack* and *buffer*
- Initialize the *stack* as empty and the *buffer* to contain the sentence

Incremental algorithm

- Parse structure $\rightarrow$ sequence of transitions
- Transition : **shift** and **reduce** actions
- Data structures : *stack* and *buffer*
- Initialize the *stack* as empty and the *buffer* to contain the sentence
- At each time step, track:
  - Data structure contents (*parser state*)
  - History of transitions

Incremental algorithm

- Parse structure $\rightarrow$ sequence of transitions
- Transition : **shift** and **reduce** actions
- Data structures : *stack* and *buffer*
- Initialize the *stack* as empty and the *buffer* to contain the sentence
- At each time step, track:
  - Data structure contents (*parser state*)
  - History of transitions
- Terminate when the *buffer* is empty

Incremental algorithm

- Parse structure $\rightarrow$ sequence of transitions
- Transition : **shift** and **reduce** actions
- Data structures : *stack* and *buffer*
- Initialize the *stack* as empty and the *buffer* to contain the sentence
- At each time step, track:
  - Data structure contents (*parser state*)
  - History of transitions
- Terminate when the *buffer* is empty

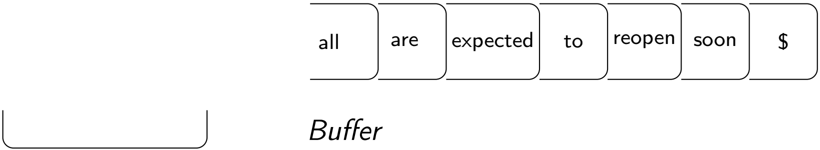
Modified arc-eager algorithm [Nivre, 2008, Henderson et al., 2008, Henderson et al., 2013, Gesmundo et al., 2009, Titov et al., 2009]

Transitions for syntax

Transitions for syntax

- S-Shift
-
-
-

all are expected to reopen soon



Buffer

Stack

Transitions for syntax

- S-Shift ✓
-
-
-

all are expected to reopen soon

are	expected	to	reopen	soon	$

Buffer

all

Stack

Transitions for syntax

- S-Shift
- S-Left
-
-

all are expected to reopen soon

are	expected	to	reopen	soon	$

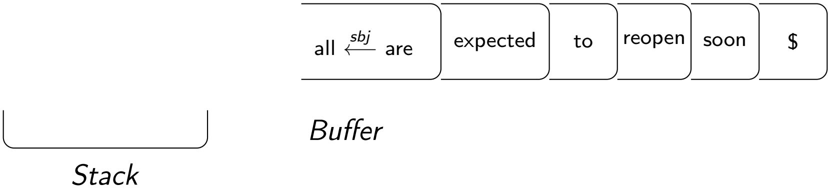
Buffer

all

Stack

Transitions for syntax

- S-Shift
- S-Left ✓
-
-



Transitions for syntax

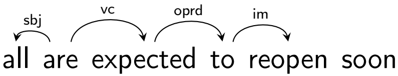
- S-Shift
- S-Left ✓
-
-

[are] | expected | to | reopen | soon | $

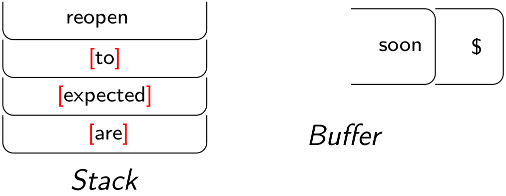
Stack

Buffer

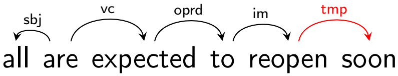
Transitions for syntax



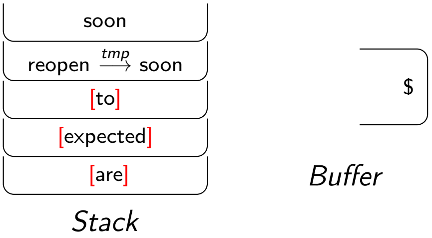
- S-Shift
- S-Left
- S-Right
-



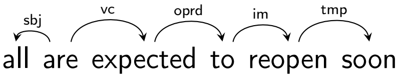
Transitions for syntax

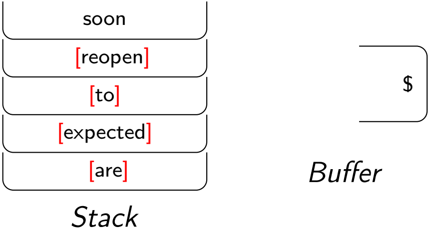


- S-Shift
- S-Left
- S-Right ✓
-



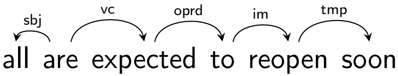
Transitions for syntax



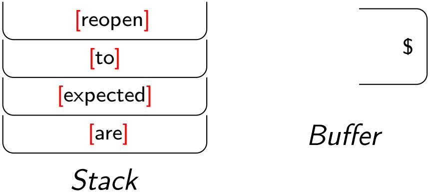


- S-Shift
- S-Left
- S-Right
- S-Reduce

Transitions for syntax



- S-Shift
- S-Left
- S-Right
- S-Reduce ✓



More transitions for semantics

- M-Shift
- M-Left
- M-Right
- M-Reduce

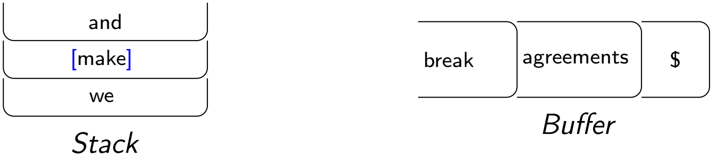
More transitions for semantics

- M-Pred
-
-

we make and break agreements

make.03

A0



More transitions for semantics

- M-Pred ✓
-
-

we make and break agreements

make.03 **break.01**

A0

and

[make]

we

Stack

break **break.01** | agreements | $

Buffer

we make and break agreements

make.03 **break.01**

A0

- M-Pred
- M-Swap
-

More transitions for semantics

we make and break agreements

make.03 **break.01**

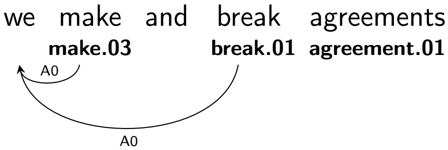
A0

- M-Pred
- M-Swap ✓
-

Stack
we
[make]

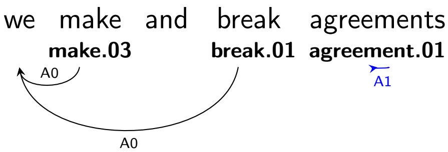
Buffer		
[break]	agreements	$

More transitions for semantics

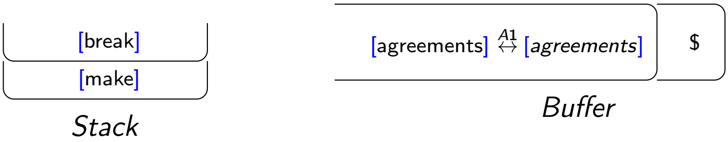


- M-Pred
- M-Swap
- M-Self

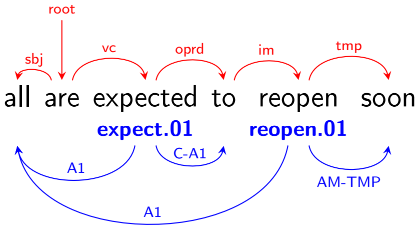
More transitions for semantics



- M-Pred
- M-Swap
- M-Self ✓

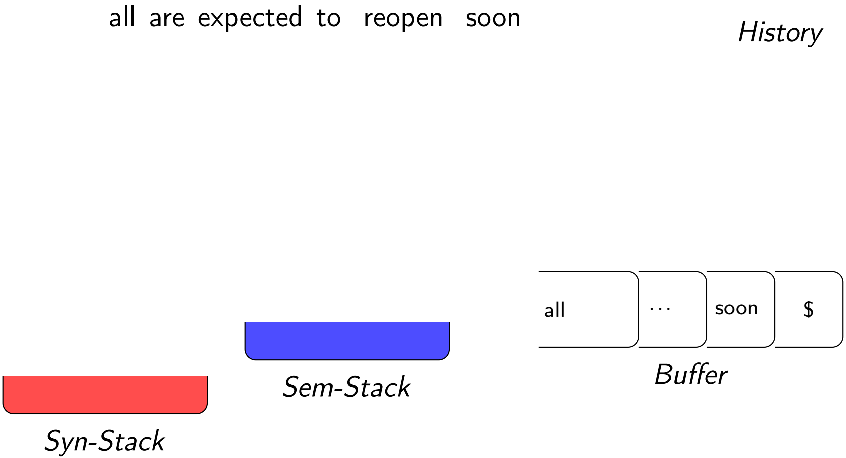


Synchronizing syntax and semantics



- Two stacks: *Syn-Stack* and *Sem-Stack*
- Share the *Buffer*
- Syntactic transitions followed by semantic transitions for a given *Buffer* state [Henderson et al., 2008]

An example transition sequence



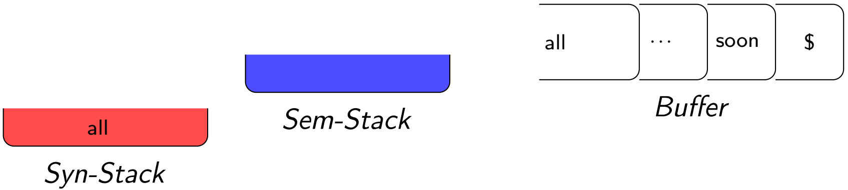
[x] denotes parse substructure headed by x

An example transition sequence

all are expected to reopen soon

History

S-Shift



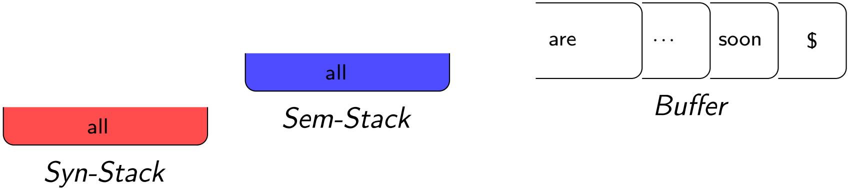
[x] denotes parse substructure headed by x

An example transition sequence

all are expected to reopen soon

History

S-Shift
M-Shift



[x] denotes parse substructure headed by x

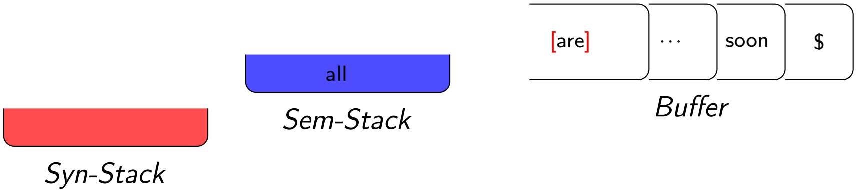
An example transition sequence

sbj

all are expected to reopen soon

History

S-Shift
M-Shift
S-Left(sbj)



[x] denotes parse substructure headed by x

An example transition sequence

sbj

all are expected to reopen soon

History

S-Shift
M-Shift
S-Left(sbj)
S-Shift

[are] | ··· | soon | $

Buffer

all

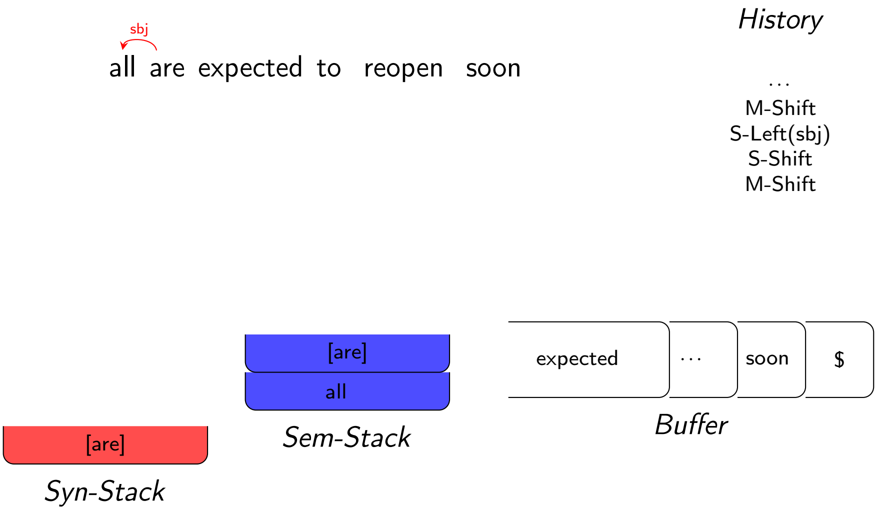
Sem-Stack

[are]

Syn-Stack

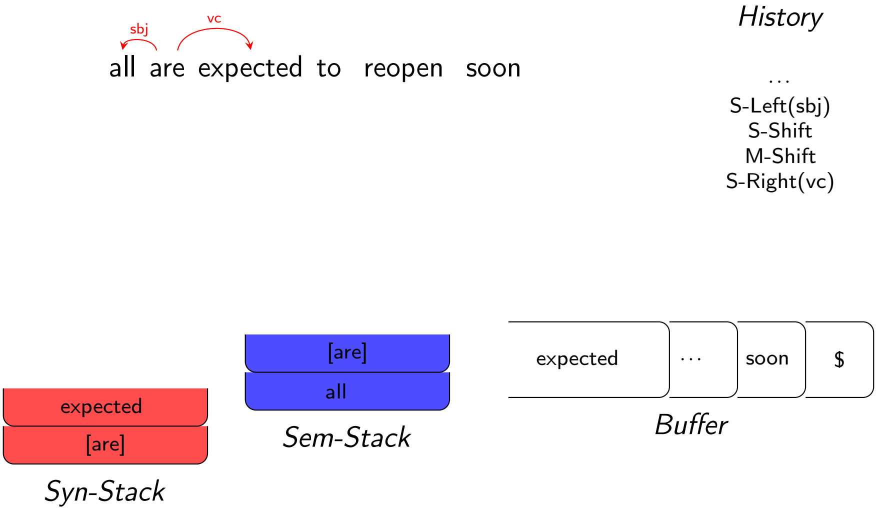
[x] denotes parse substructure headed by x

An example transition sequence



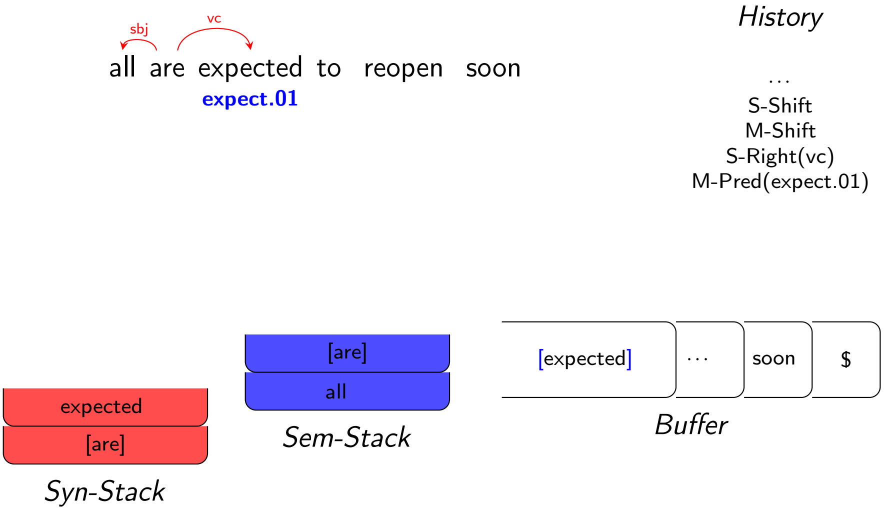
[x] denotes parse substructure headed by x

An example transition sequence



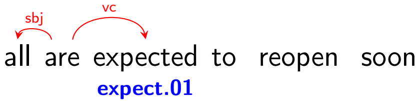
[x] denotes parse substructure headed by x

An example transition sequence



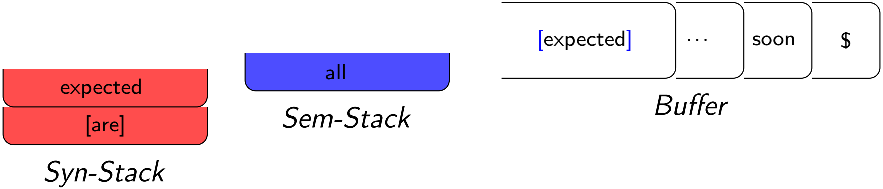
[x] denotes parse substructure headed by x

An example transition sequence



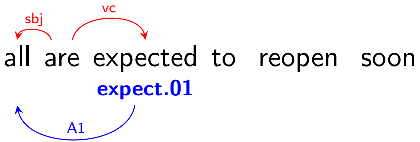
History

...
M-Shift
S-Right(vc)
M-Pred(expect.01)
M-Reduce



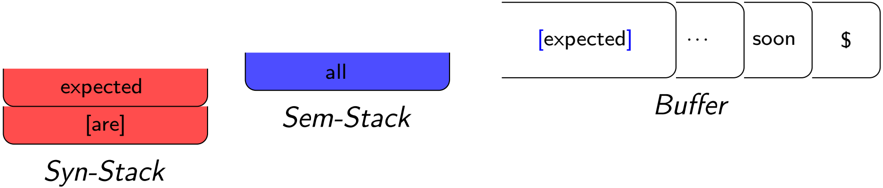
[x] denotes parse substructure headed by x

An example transition sequence



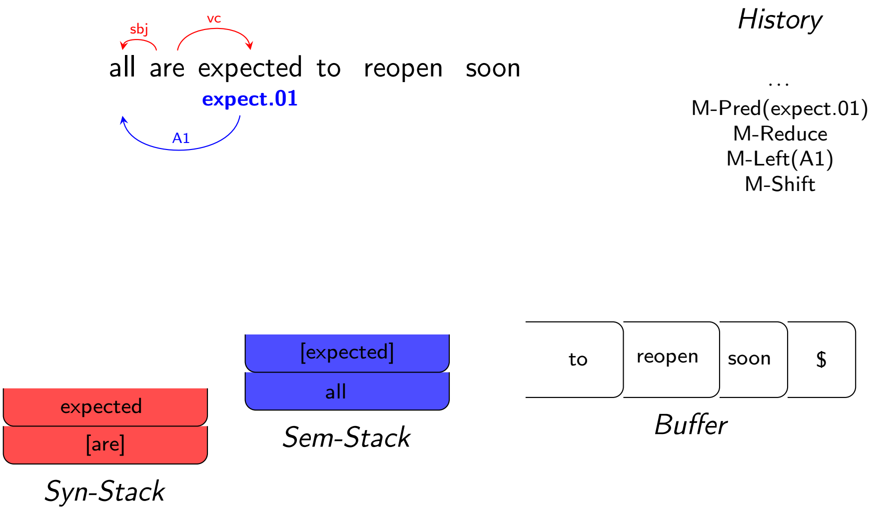
History

...
S-Right(vc)
M-Pred(expect.01)
M-Reduce
M-Left(A1)



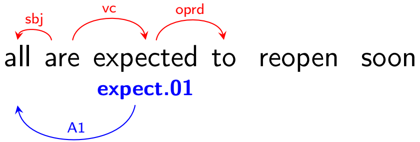
[x] denotes parse substructure headed by x

An example transition sequence



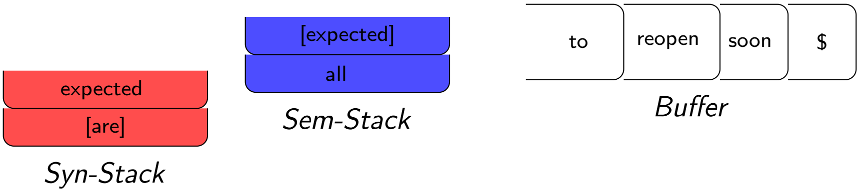
[x] denotes parse substructure headed by x

An example transition sequence



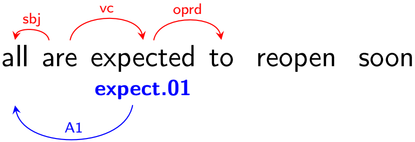
History

...
M-Reduce
M-Left(A1)
M-Shift



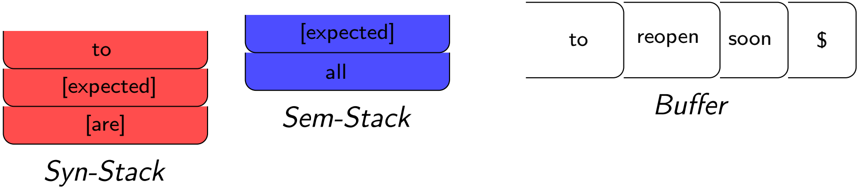
[x] denotes parse substructure headed by x

An example transition sequence



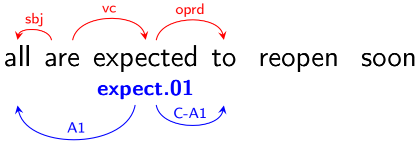
History

...
M-Left(A1)
M-Shift
S-Right(oprd)



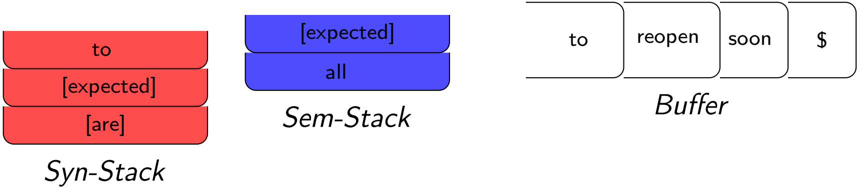
[x] denotes parse substructure headed by x

An example transition sequence



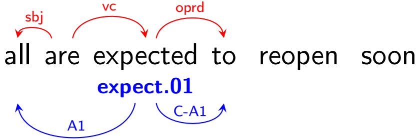
History

...
M-Shift
S-Right(oprd)
M-Right(C-A1)



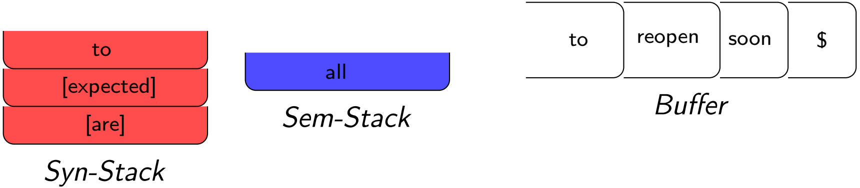
[x] denotes parse substructure headed by x

An example transition sequence



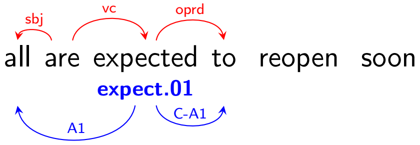
History

...
S-Right(oprd)
M-Right(C-A1)
M-Reduce



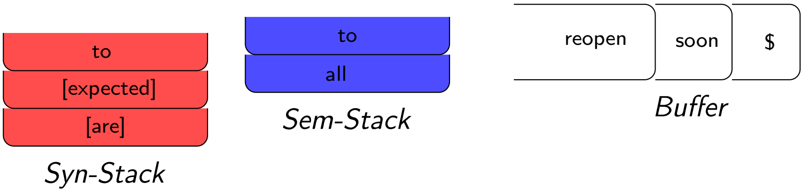
[x] denotes parse substructure headed by x

An example transition sequence



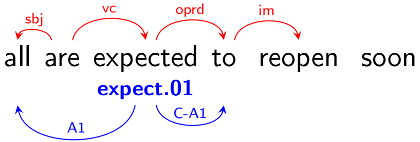
History

...
M-Right(C-A1)
M-Reduce
M-Shift



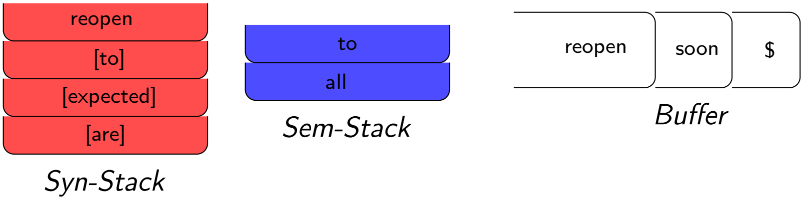
[x] denotes parse substructure headed by x

An example transition sequence



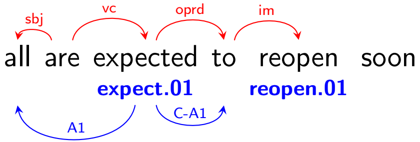
History

...
M-Reduce
M-Shift
S-Right(im)



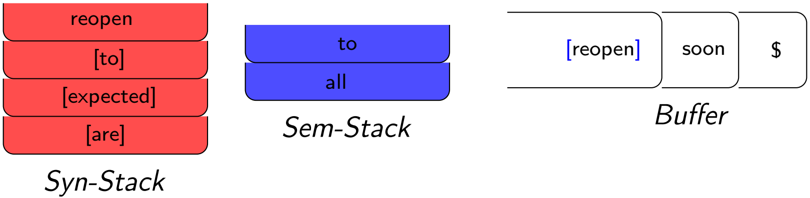
[x] denotes parse substructure headed by x

An example transition sequence



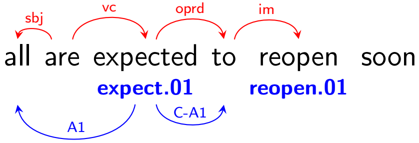
History

...
M-Shift
S-Right(im)
M-Pred(reopen.01)



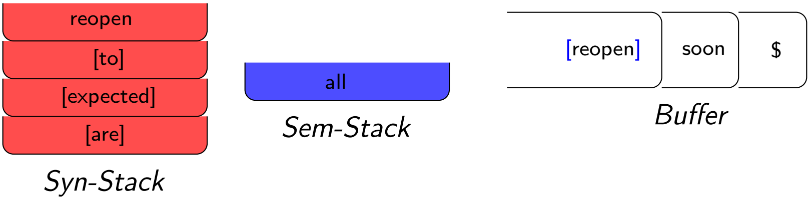
[x] denotes parse substructure headed by x

An example transition sequence



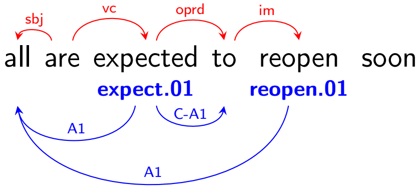
History

...
S-Right(im)
M-Pred(reopen.01)
M-Reduce



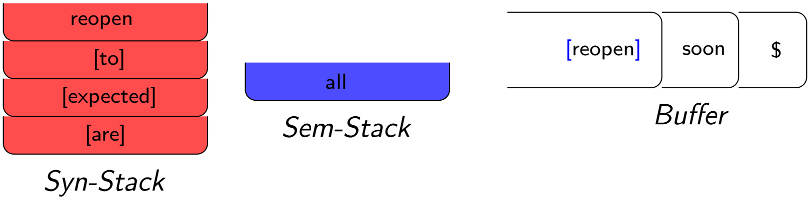
[x] denotes parse substructure headed by x

An example transition sequence



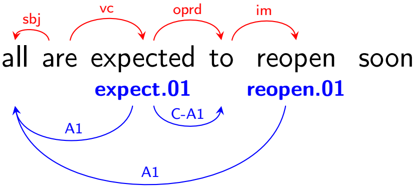
History

...
M-Pred(reopen.01)
M-Reduce
M-Left(A1)



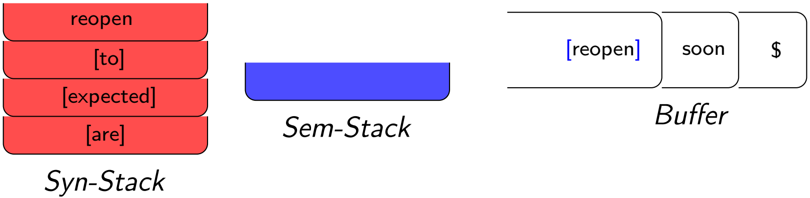
[x] denotes parse substructure headed by x

An example transition sequence



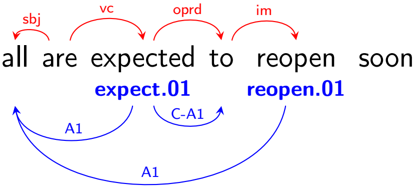
History

...
M-Reduce
M-Left(A1)
M-Reduce



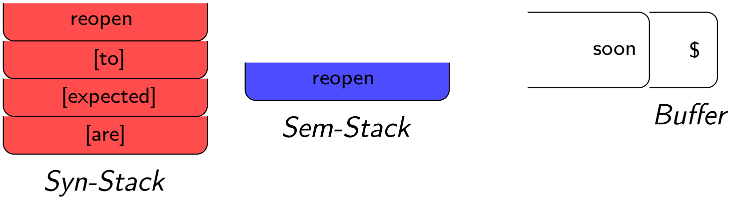
[x] denotes parse substructure headed by x

An example transition sequence



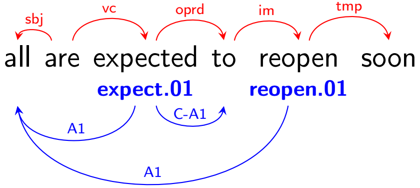
History

...
M-Left(A1)
M-Reduce
M-Shift



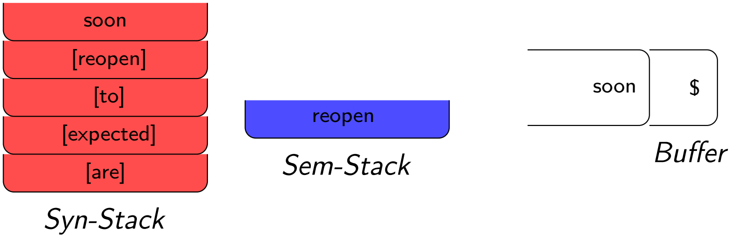
[x] denotes parse substructure headed by x

An example transition sequence



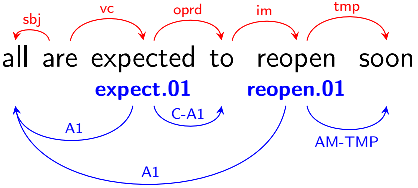
History

...
M-Reduce
M-Shift
S-Right(tmp)



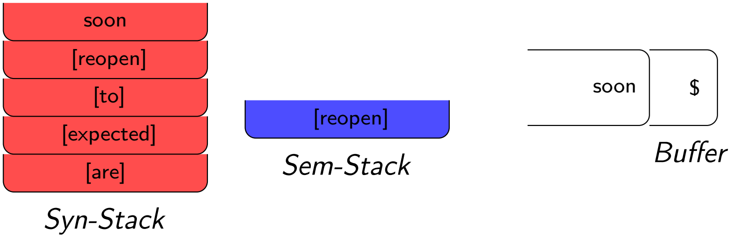
[x] denotes parse substructure headed by x

An example transition sequence



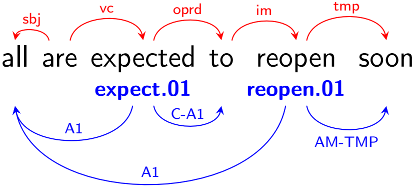
History

...
M-Shift
S-Right(tmp)
M-Right(AM-TMP)



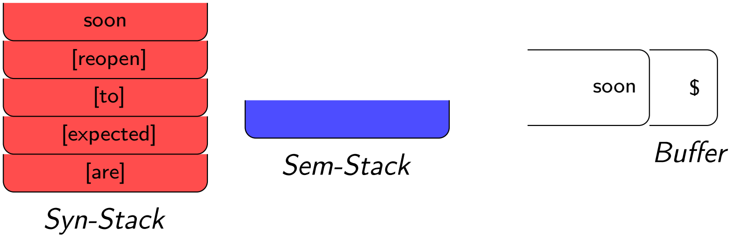
[x] denotes parse substructure headed by x

An example transition sequence



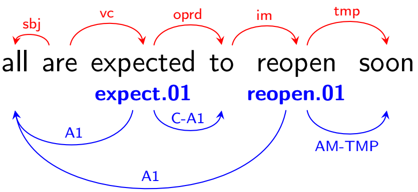
History

...
S-Right(tmp)
M-Right(AM-TMP)
M-Reduce



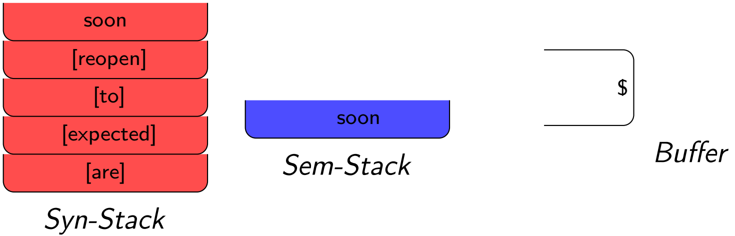
[x] denotes parse substructure headed by x

An example transition sequence



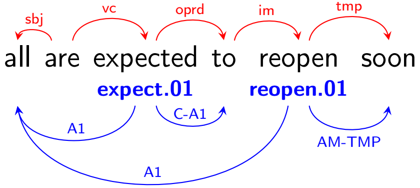
History

...
M-Right(AM-TMP)
M-Reduce
M-Shift



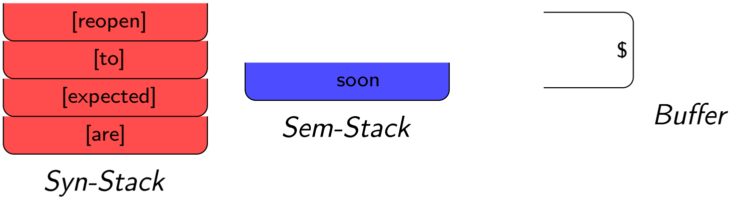
[x] denotes parse substructure headed by x

An example transition sequence



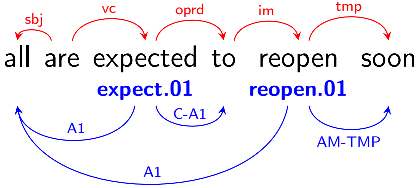
History

…
M-Reduce
M-Shift
S-Reduce



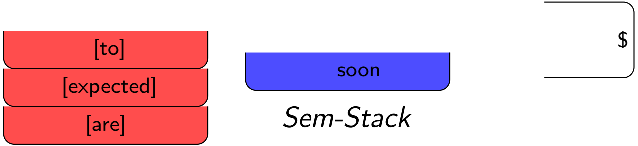
[x] denotes parse substructure headed by x

An example transition sequence



History

...
M-Shift
S-Reduce
S-Reduce



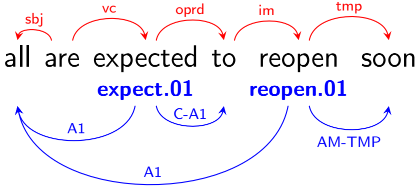
Syn-Stack

Sem-Stack

Buffer

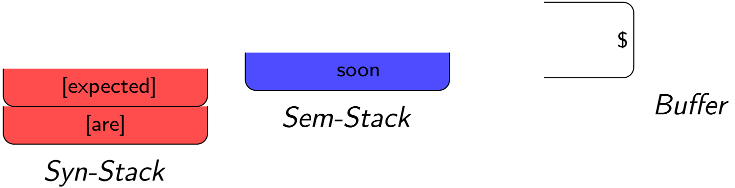
[x] denotes parse substructure headed by x

An example transition sequence



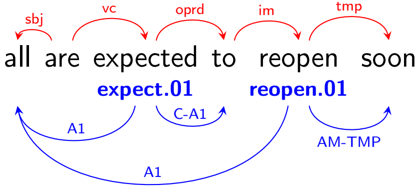
History

...
S-Reduce
S-Reduce
S-Reduce



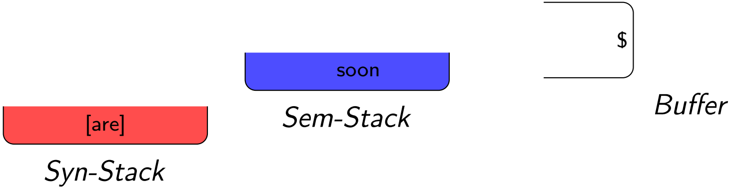
[x] denotes parse substructure headed by x

An example transition sequence



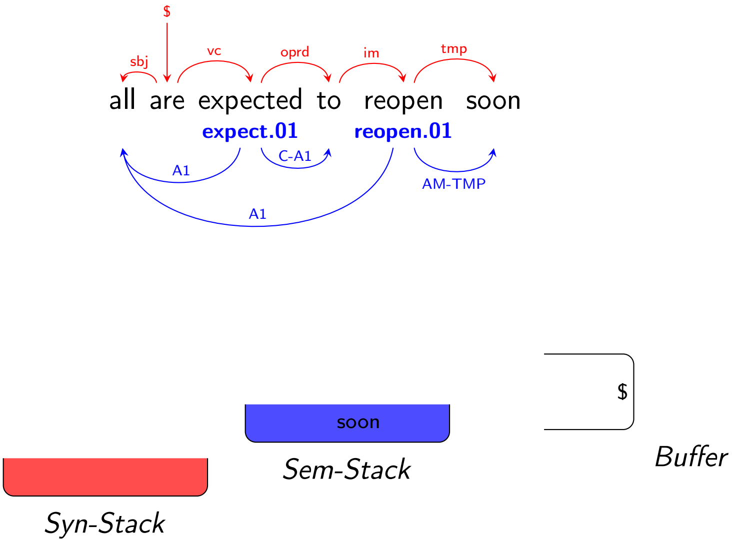
History

...
S-Reduce
S-Reduce
S-Reduce



[x] denotes parse substructure headed by x

An example transition sequence

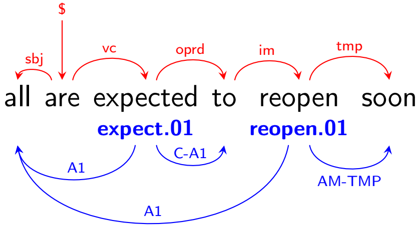


History

...
S-Reduce
S-Reduce
S-Left($)

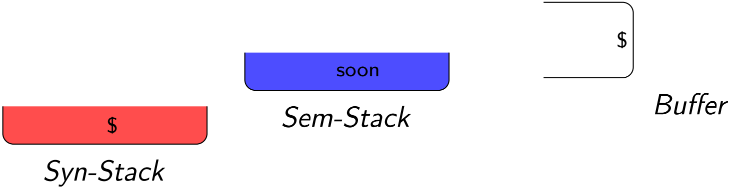
[x] denotes parse substructure headed by x

An example transition sequence



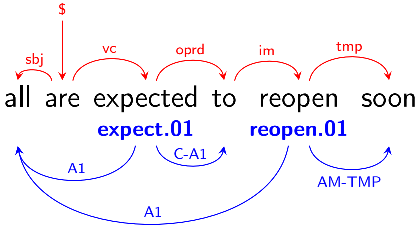
History

...
S-Reduce
S-Left($)
S-Shift



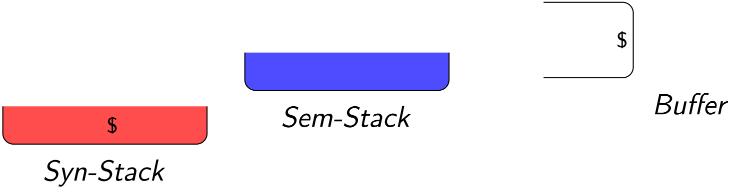
[x] denotes parse substructure headed by x

An example transition sequence



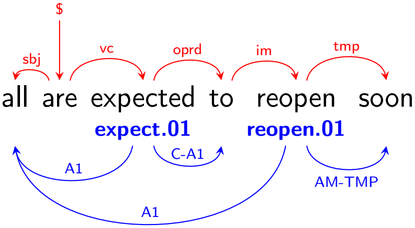
History

...
S-Left($)
S-Shift
M-Reduce



[x] denotes parse substructure headed by x

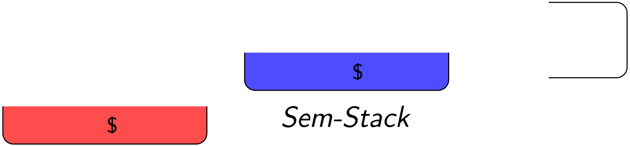
An example transition sequence



History

...
S-Left($)
S-Shift
M-Reduce
M-Shift

Linear algorithm



Sem-Stack

Buffer

Syn-Stack

[x] denotes parse substructure headed by x

Outline

Introduction

Incremental Algorithm

Stack LSTM Model

CoNLL 2008-09 Shared Task Results

Stack LSTM Model

- LSTMs: Recurrent neural networks with special memory cell [Hochreiter and Schmidhuber, 1997, Graves, 2013] to learn fixed-size representations for variable-length sequences

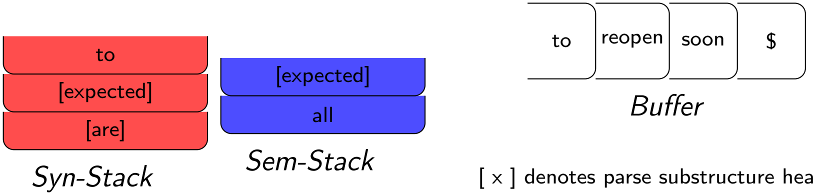
Stack LSTM Model

- LSTMs: Recurrent neural networks with special memory cell [Hochreiter and Schmidhuber, 1997, Graves, 2013] to learn fixed-size representations for variable-length sequences
- **Stack LSTMs**: LSTMs equipped with stack operations [Dyer et al., 2015]
  - summary $\rightarrow$ return a fixed-size continuous representation
  - push $\rightarrow$ add to the sequence
  - pop $\rightarrow$ remove from the sequence

Stack LSTM for Joint Parsing

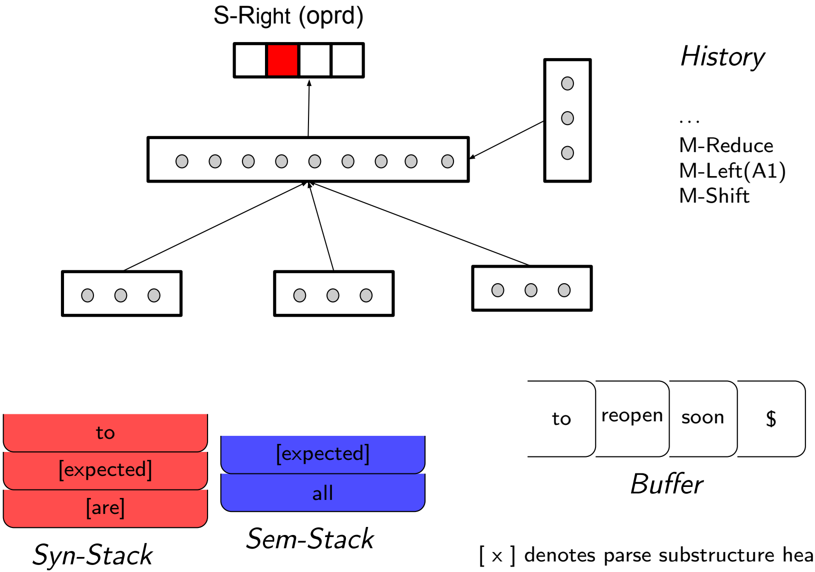
History

...
M-Reduce
M-Left(A1)
M-Shift



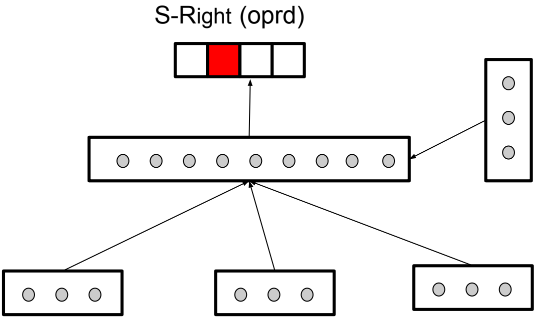
[x] denotes parse substructure headed by x

Stack LSTM for Joint Parsing



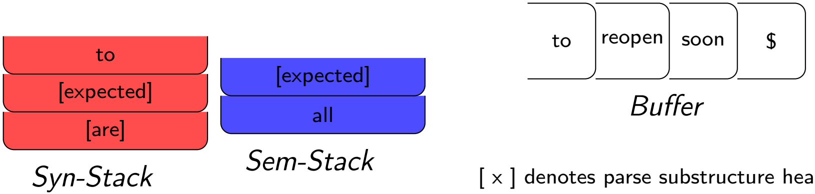
[x] denotes parse substructure headed by x

Stack LSTM for Joint Parsing



History

. . .
M-Reduce
M-Left(A1)
M-Shift



[x] denotes parse substructure headed by x

Outline

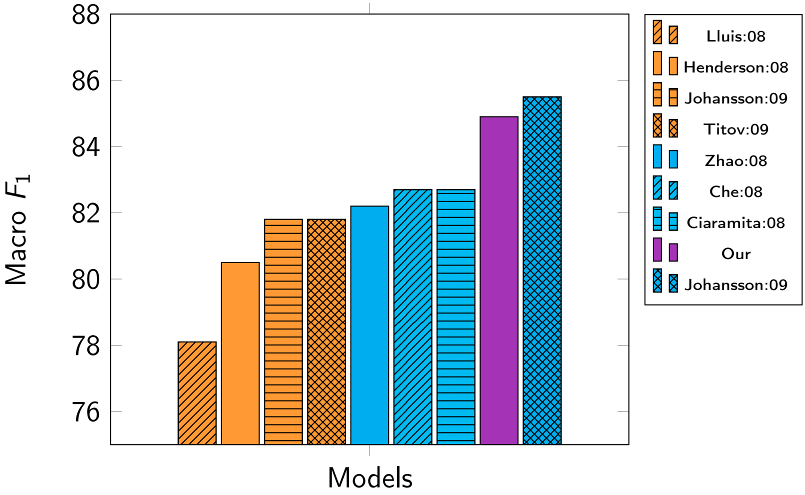
CoNLL Shared Tasks

- 2008: English only
- 2009: Multilingual
- Evaluation metrics:
  - Syntax: Labeled Accuracy Score (LAS)
  - SRL: Semantic F_1
  - Rank systems: Macro F_1

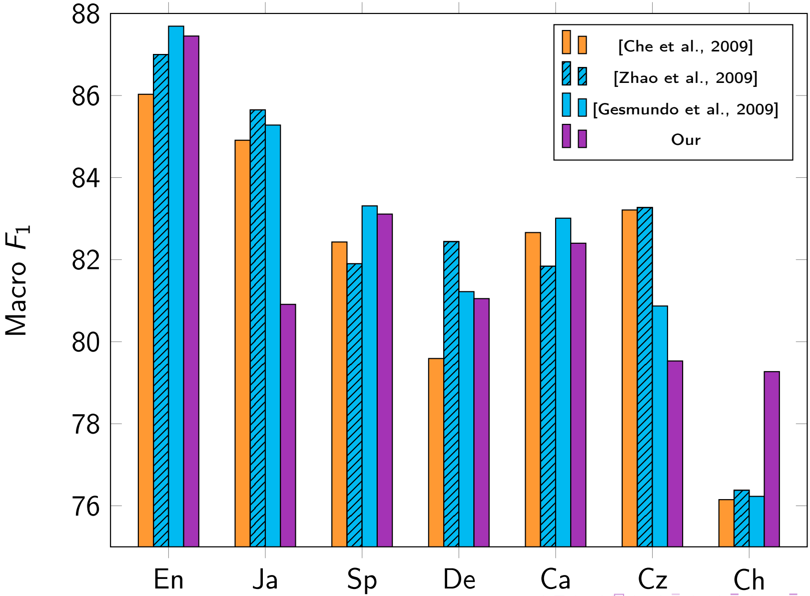
Experimental Setup

- POS tags were used, but no other provided features
- No manually-designed features
- Dataset-specific hyperparameter tuning

CoNLL 2008 (English only) Shared Task



CoNLL 2009 (Multilingual) Shared Task



Conclusion

Take-aways!

- Incremental algorithm (linear) + model using stack LSTMs
- Avoid the effort involved in manual feature engineering
- Light-weight alternative to expensive pipelined systems

Code available at
`https://github.com/clab/joint-lstm-parser`

References I

Björkelund, A., Bohnet, B., Hafdell, L., and Nugues, P. (2010). A high-performance syntactic and semantic dependency parser. In *Proc. of COLING*.

Che, W., Li, Z., Hu, Y., Li, Y., Qin, B., Liu, T., and Li, S. (2008). A cascaded syntactic and semantic dependency parsing system. In *Proc. of CoNLL*.

Che, W., Li, Z., Li, Y., Guo, Y., Qin, B., and Liu, T. (2009). Multilingual dependency-based syntactic and semantic parsing. In *Proc. of CoNLL*.

Ciaramita, M., Attardi, G., Dell'Orletta, F., and Surdeanu, M. (2008). DeSRL: A linear-time semantic role labeling system. In *Proc. of CoNLL*.

Dyer, C., Ballesteros, M., Ling, W., Matthews, A., and Smith, N. A. (2015). Transition-based dependency parsing with stack long short-term memory. In *Proc. of ACL*.

FitzGerald, N., Täckström, O., Ganchev, K., and Das, D. (2015). Semantic role labelling with neural network factors. In *Proc. of EMNLP*.

Gesmundo, A., Henderson, J., Merlo, P., and Titov, I. (2009). A latent variable model of synchronous syntactic-semantic parsing for multiple languages. In *Proc. of CoNLL*.

Graves, A. (2013). Generating sequences with recurrent neural networks. arXiv:1308.0850.

He, H., Daumé III, H., and Eisner, J. (2013). Dynamic feature selection for dependency parsing. In *Proc. of EMNLP*.

Henderson, J., Merlo, P., Musillo, G., and Titov, I. (2008). A latent variable model of synchronous parsing for syntactic and semantic dependencies. In *Proc. of CoNLL*.

Henderson, J., Merlo, P., Titov, I., and Musillo, G. (2013). Multi-lingual joint parsing of syntactic and semantic dependencies with a latent variable model. *Computational Linguistics*, 39(4):949–998.

References II

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Johansson, R. (2009). Statistical bistratal dependency parsing. In *Proc. of EMNLP*.

Lei, T., Zhang, Y., i Villodre, L. M., Moschitti, A., and Barzilay, R. (2015). High-order low-rank tensors for semantic role labeling. In *Proc. of NAACL*.

Levin, B. and Hovav, M. R. (1996). Lexical semantics and syntactic structure. *The handbook of contemporary semantic theory*, 18:487–507.

Lluís, X. and Màrquez, L. (2008). A joint model for parsing syntactic and semantic dependencies. In *Proc. of CoNLL*.

Nivre, J. (2008). Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34(4):513–553.

Palmer, M., Gildea, D., and Kingsbury, P. (2005). The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Roth, M. and Lapata, M. (2016). Neural semantic role labeling with dependency path embeddings. arXiv:1605.07515.

Roth, M. and Woodsend, K. (2014). Composition of word representations improves semantic role labelling. In *Proc. of EMNLP*.

Täckström, O., Ganchev, K., and Das, D. (2015). Efficient inference and structured learning for semantic role labeling. *Transactions of the ACL*, 3:29–41.

Titov, I., Henderson, J., Merlo, P., and Musillo, G. (2009). Online graph planarisation for synchronous parsing of semantic and syntactic dependencies. In *Proc. of IJCAI*.

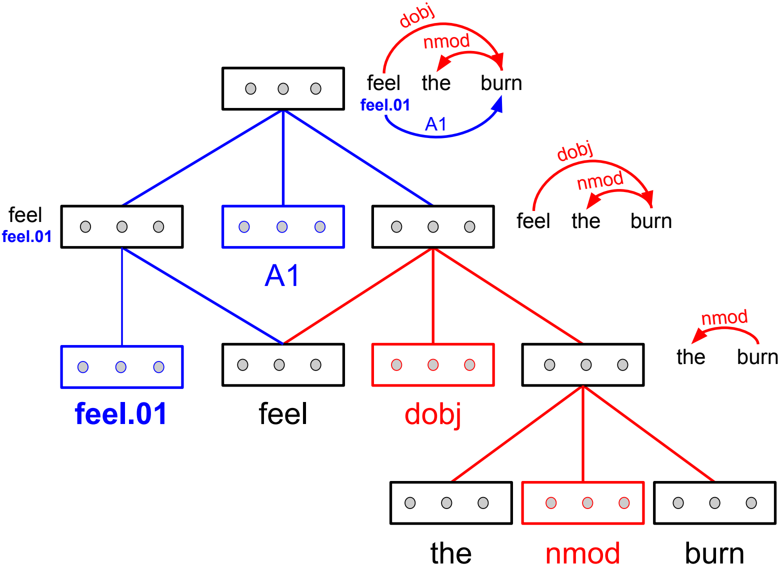
References III

Zhao, H., Chen, W., Kazama, J., Uchimoto, K., and Torisawa, K. (2009). Multilingual dependency learning: Exploiting rich features for tagging syntactic and semantic dependencies. In *Proc. of CoNLL*.

Zhao, H. and Kit, C. (2008). Parsing syntactic and semantic dependencies with two single-stage maximum entropy models. In *Proc. of CoNLL*.

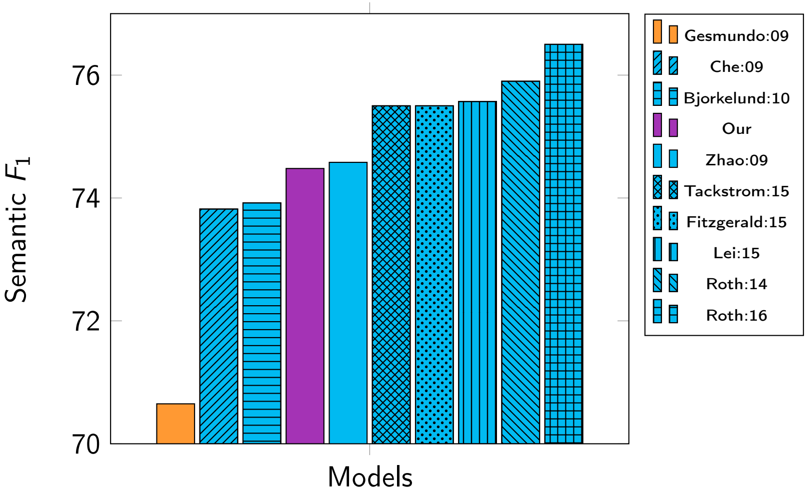
Extras

Syntactic-semantic composition

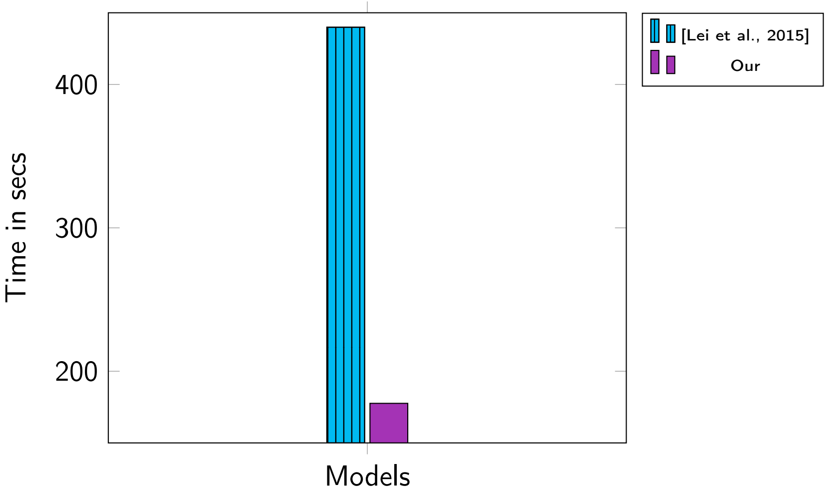


SRL performance on out-of-domain (Brown) data

CoNLL 2009 Shared Task



Time to decode the CoNLL 2009 English dataset



Experiments were run end to end on a single CPU