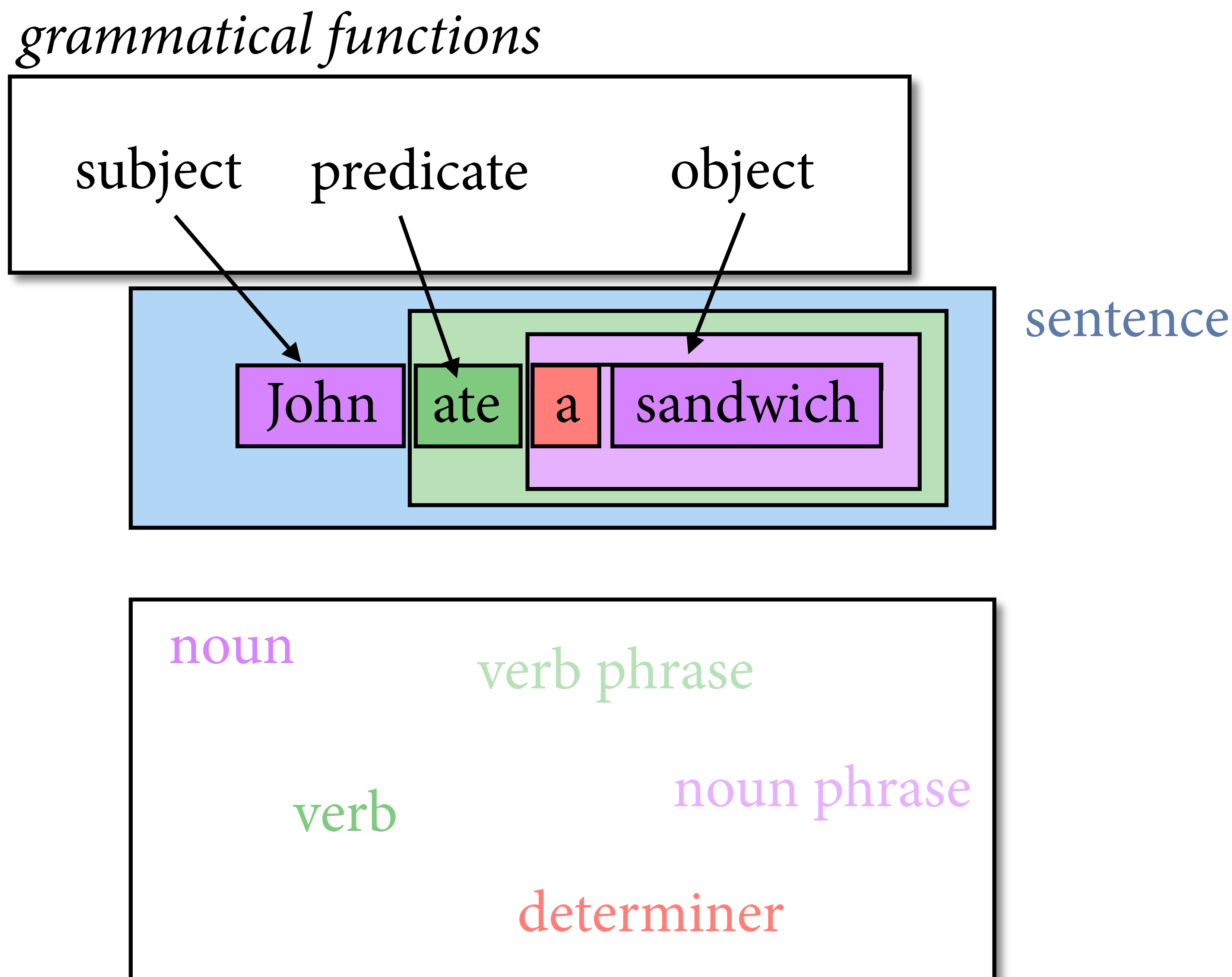
Neural and neurosymbolic parsing

Computational Linguistics

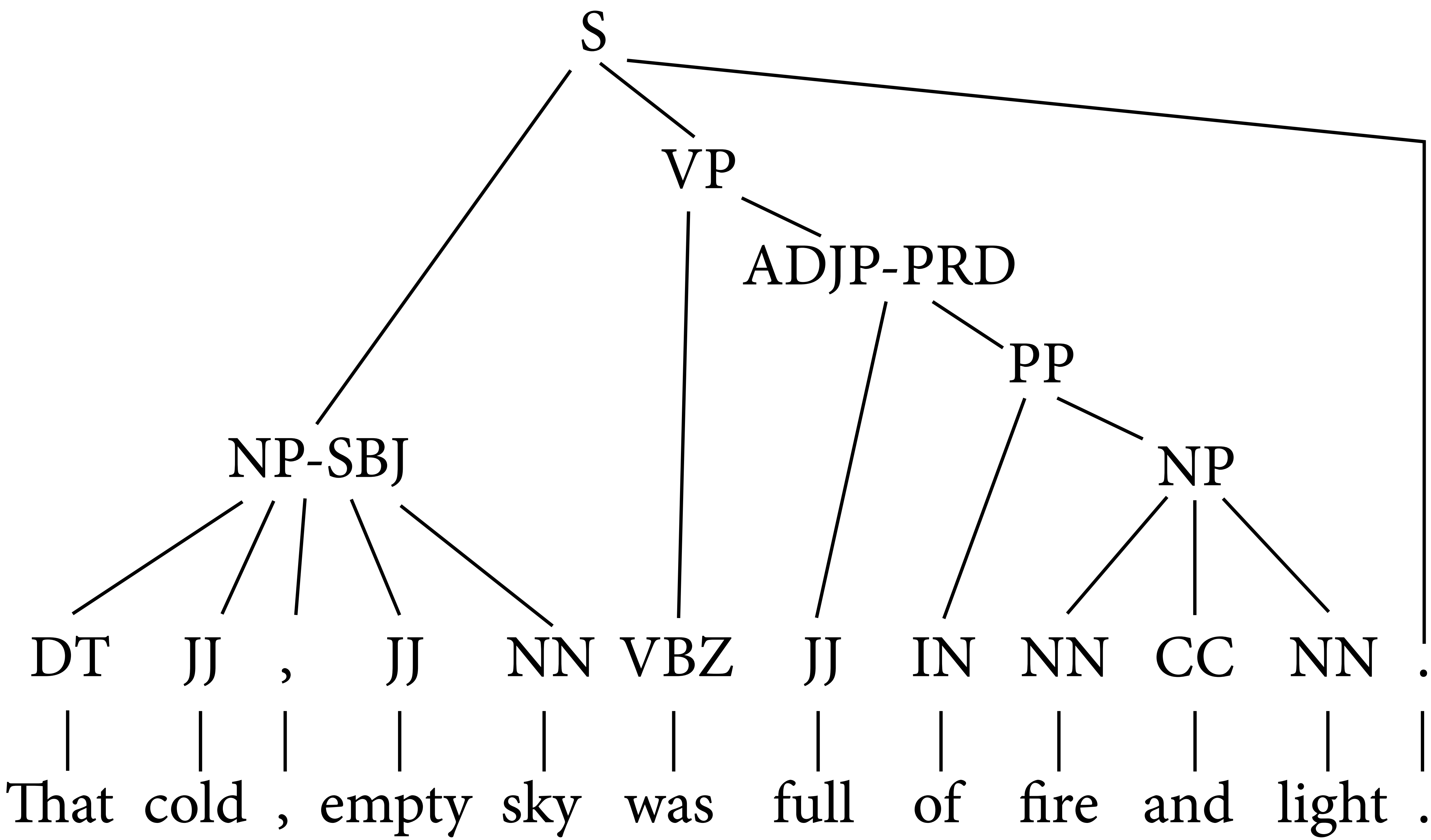
Alexander Koller

12 December 2023

Sentences have structure



Penn Treebank

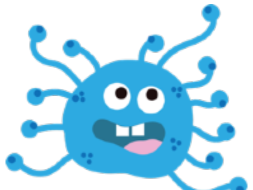


Parsing with neural models

- The syntactic structure of a sentence is a tree.
  So we need models of syntax that know
  what a tree is.

- Right?

- Right??



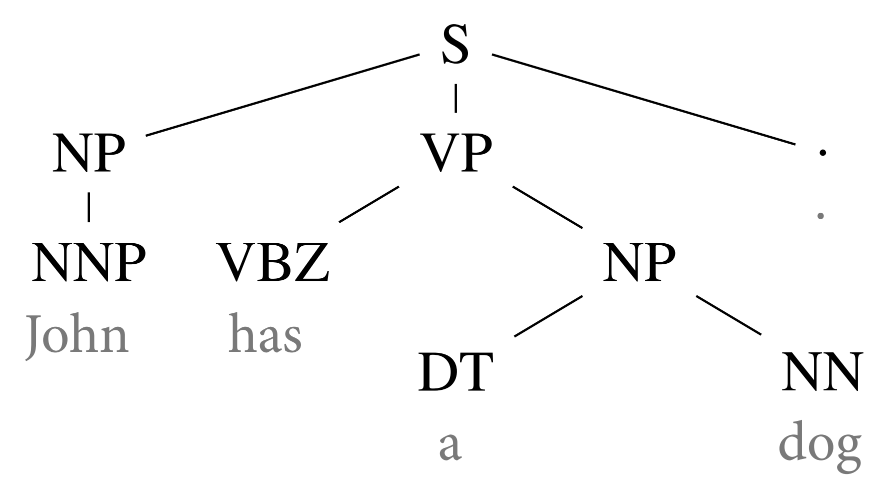
0/3

Trees as sequences

- Every tree can be uniquely and reversibly encoded as a sequence of symbols.

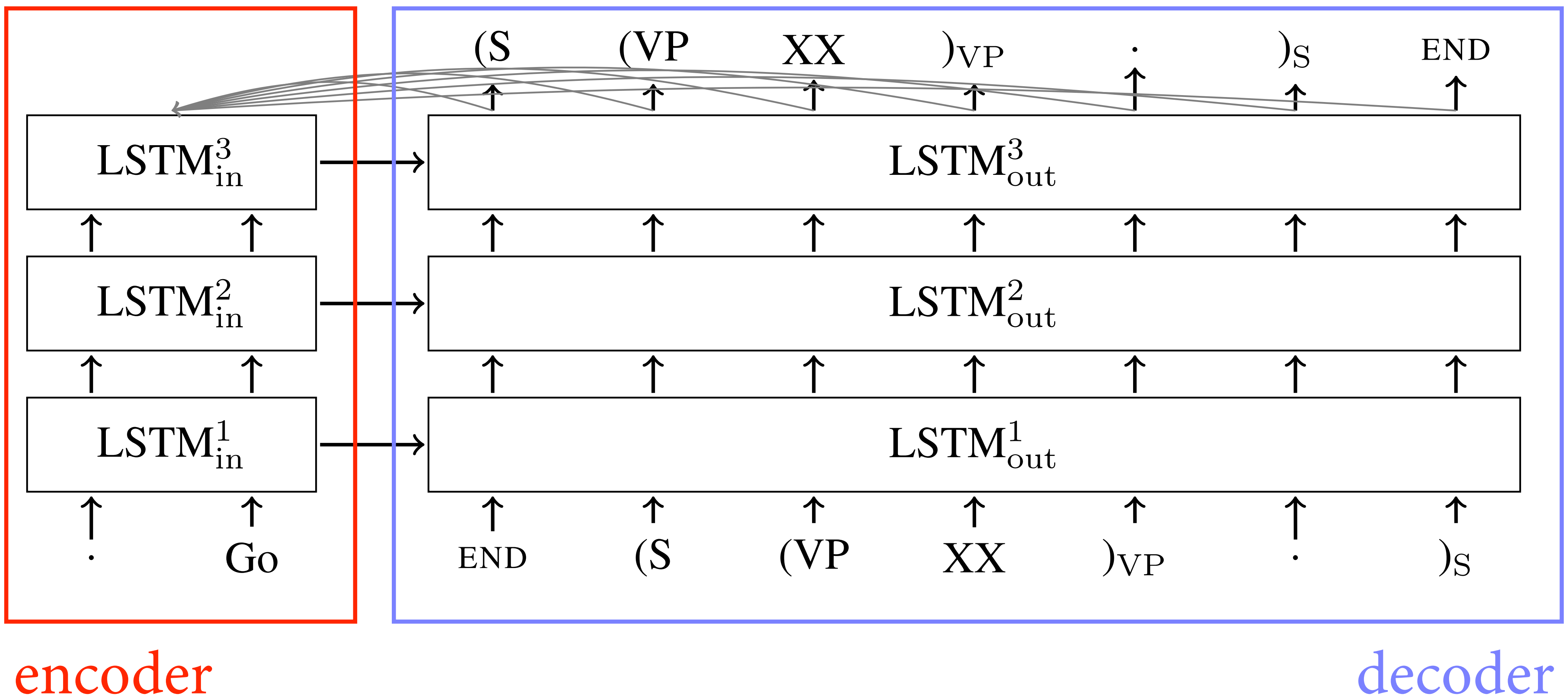
- Thus, if we know how to map sequences to sequences, we can also map sequences to trees.

input sequence:
John has a dog .



output sequence:
(S (NP NNP $)_{NP}$ (VP VBZ (NP DT NN $)_{NP}$ $)_{VP}$. $)_{S}$

Seq2seq model for parsing



Note: Input sequence is presented in reverse order. This makes LSTM-based MT models more accurate and was standard practice at the time.

(Vinyals et al. 2015)

Evaluation

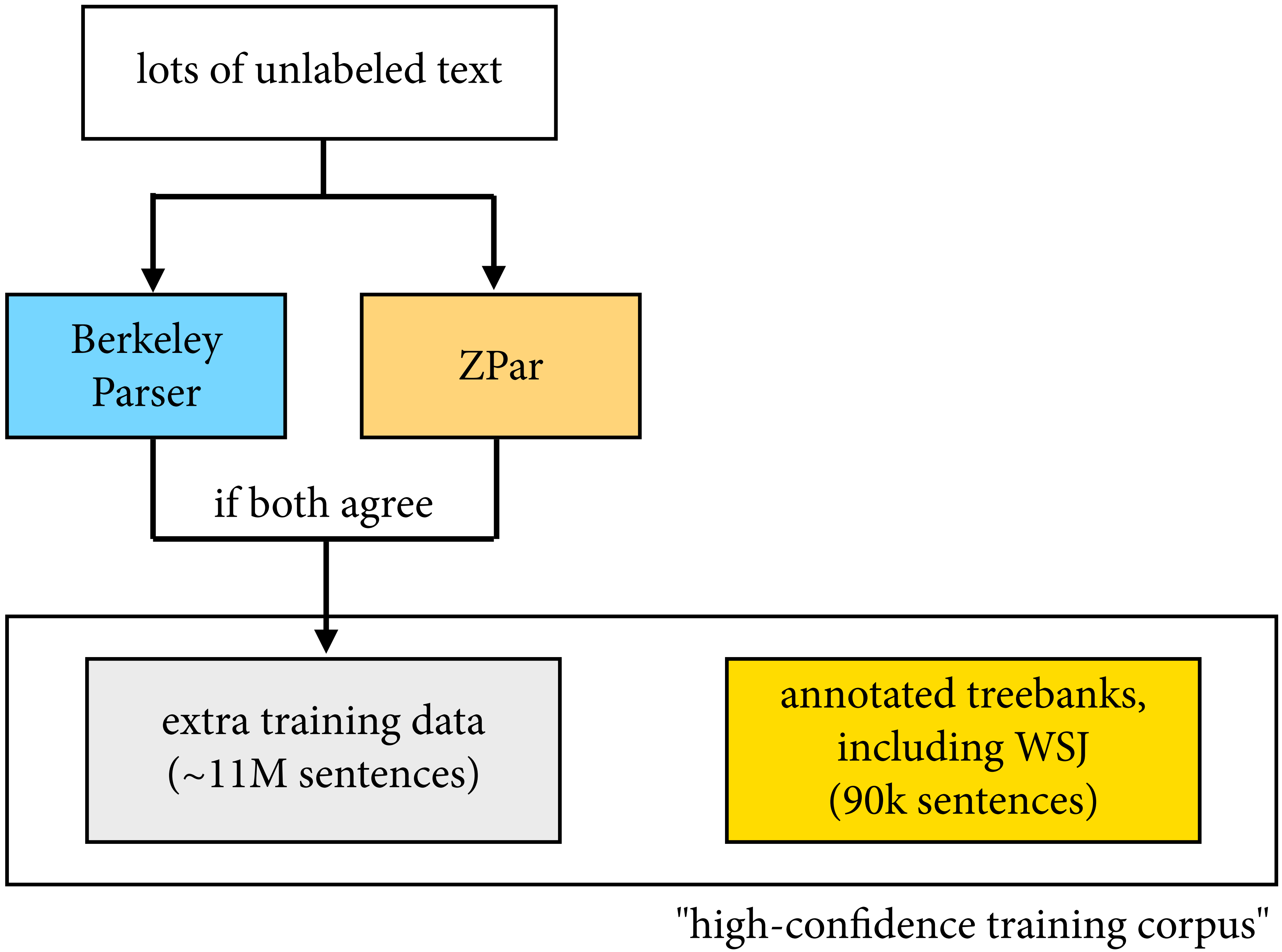
Parser	Training Set	WSJ 22	WSJ 23
baseline LSTM+D	WSJ only	< 70	< 70
LSTM+A+D	WSJ only	88.7	88.3
LSTM+A+D ensemble	WSJ only	90.7	90.5
baseline LSTM	BerkeleyParser corpus	91.0	90.5
LSTM+A	high-confidence corpus	**92.8**	**92.1**
Petrov et al. (2006) [12]	WSJ only	91.1	90.4
Zhu et al. (2013) [13]	WSJ only	N/A	90.4
Petrov et al. (2010) ensemble [14]	WSJ only	92.5	91.8
Zhu et al. (2013) [13]	semi-supervised	N/A	91.3
Huang & Harper (2009) [15]	semi-supervised	N/A	91.3
McClosky et al. (2006) [16]	semi-supervised	92.4	**92.1**

A = with cross-attention

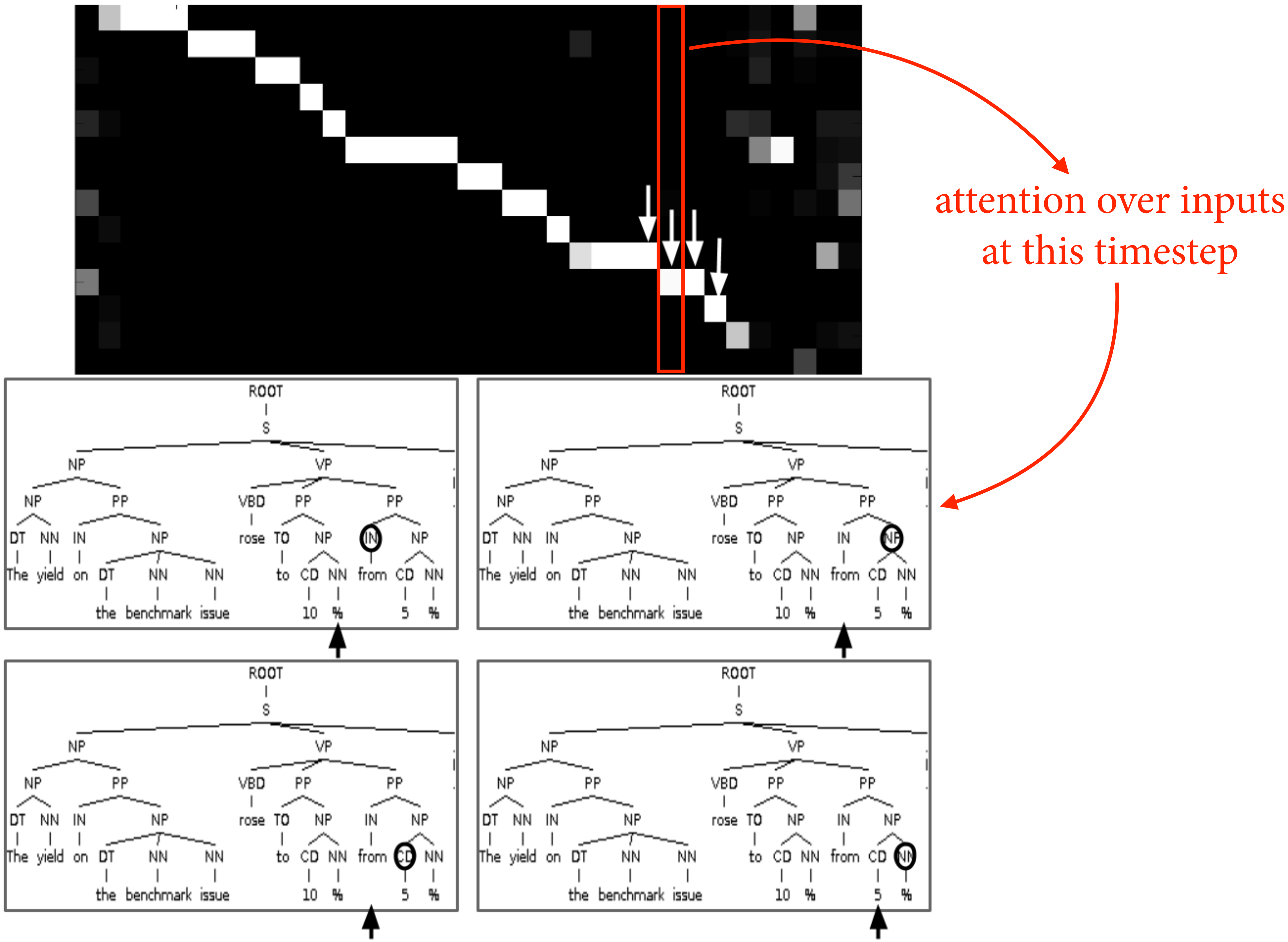
D = with dropout

> 98% of output sequences represent trees

Silver data


lots of unlabeled text
Berkeley Parser
ZPar
if both agree
extra training data (~11M sentences)
annotated treebanks, including WSJ (90k sentences)
"high-confidence training corpus"

Role of attention

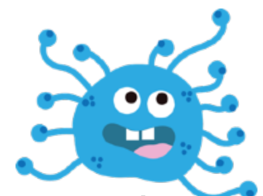


Observations

- It was shocking to me that a bare seq2seq model would do so well on a structured prediction task.

- Results on high-confidence corpus are excellent, but results on original WSJ training set only okay.

- Can we do better?



1/3

RNN Grammars

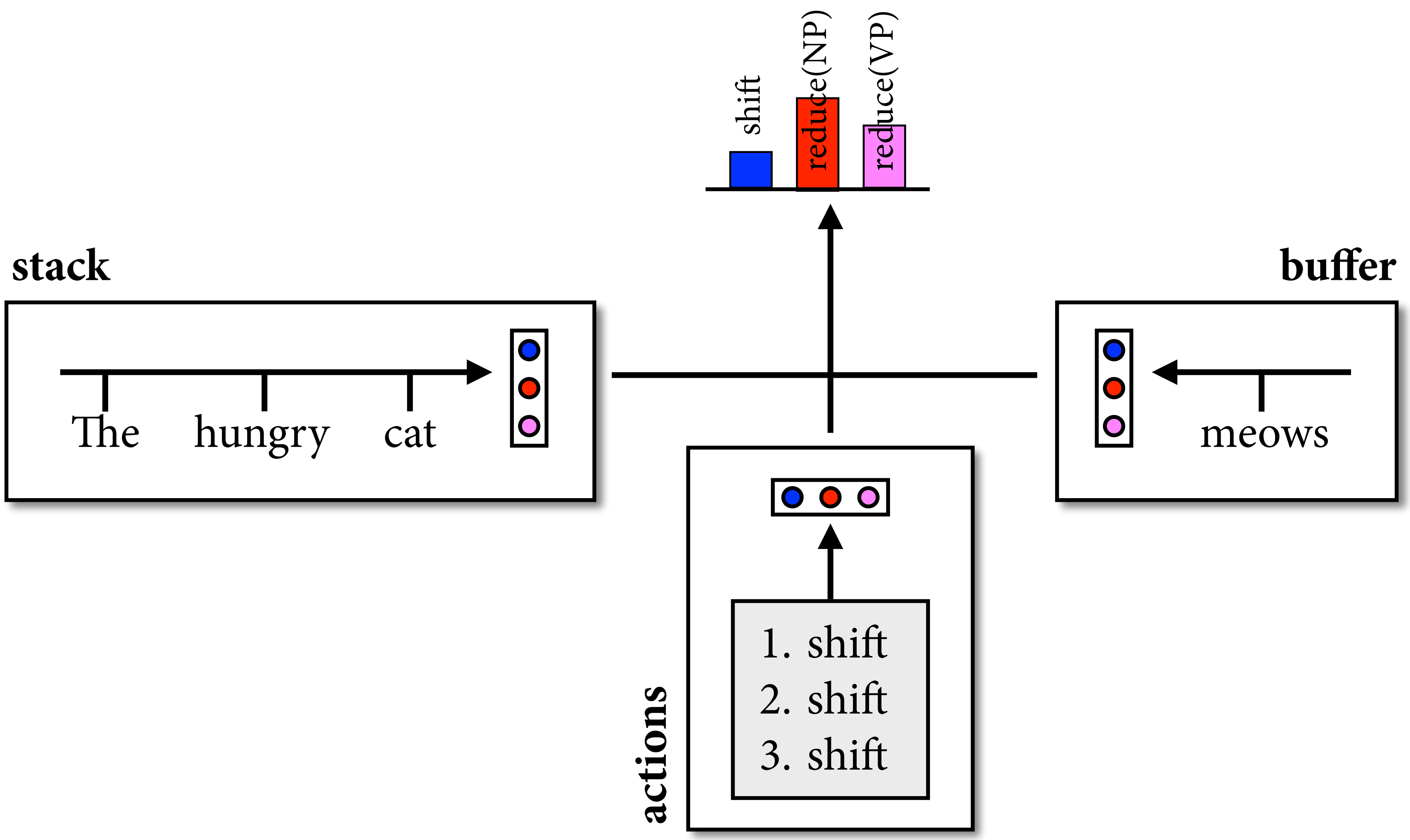
- Can we extend neural shift-reduce parsing into a parser with SOTA accuracy?

- Steps towards Recurrent Neural Network Grammars:
  - shift-reduce
  - discriminative RNNGs: add top-down context
  - generative RNNGs: better management of conditionals

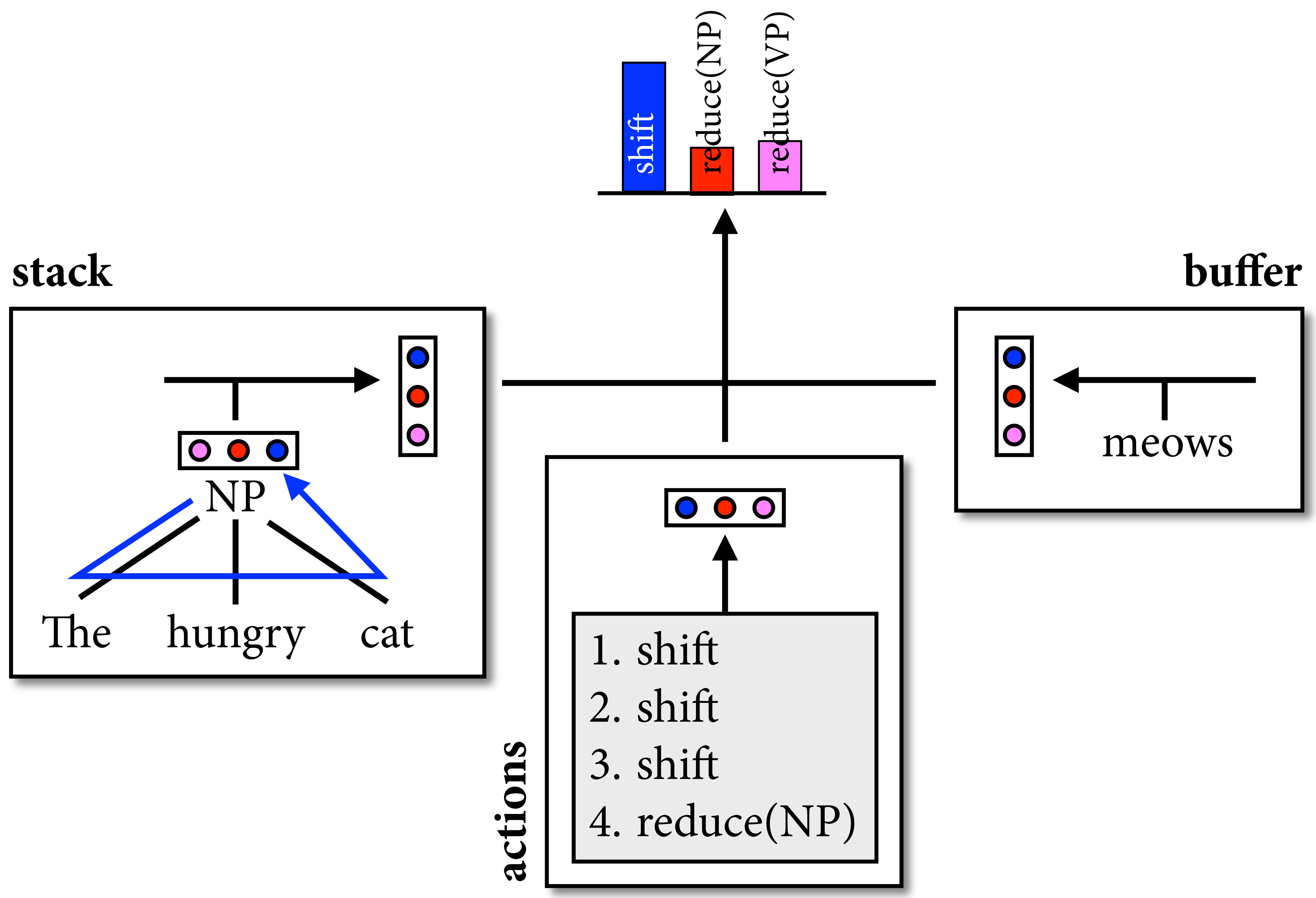
- This is a *neurosymbolic* and not a purely neural model: It has awareness of trees and uses them to compute representations.

(Dyer et al. 2016; Kuncoro et al. 2017, 2018)

Neural shift-reduce



Neural shift-reduce



(Stack LSTMs: Dyer et al. 2015)

Neural shift-reduce

- Advantages:
  - no grammar needed
  - linear runtime

- Limitations:
  - risk of incorrect early parser actions
  - no top-down context

Discriminative RNNGs

- Add top-down context:
  - add symbols for "open nonterminals" to the stack, e.g. "(S"
  - add parser action "NT(X)" to add open nonterminal "(X" to the stack.
- Picks up ideas from Earley parser for CFGs.
- Note that Vinyals parser has these "open nonterminals" too.

Example

Input: *The hungry cat meows .*

	Stack	Buffer	Action
0		*The* | *hungry* | *cat* | *meows* | .	NT(S)
1	(S	*The* | *hungry* | *cat* | *meows* | .	NT(NP)
2	(S | (NP	*The* | *hungry* | *cat* | *meows* | .	SHIFT
3	(S | (NP | *The*	*hungry* | *cat* | *meows* | .	SHIFT
4	(S | (NP | *The* | *hungry*	*cat* | *meows* | .	SHIFT
5	(S | (NP | *The* | *hungry* | *cat*	*meows* | .	REDUCE
6	(S | (NP *The hungry cat*)	*meows* | .	NT(VP)
7	(S | (NP *The hungry cat*) | (VP	*meows* | .	SHIFT
8	(S | (NP *The hungry cat*) | (VP *meows*	.	REDUCE
9	(S | (NP *The hungry cat*) | (VP *meows*)	.	SHIFT
10	(S | (NP *The hungry cat*) | (VP *meows*) | .		REDUCE
11	(S (NP *The hungry cat*) (VP *meows*) .)		

Training and parsing

- Convert parse trees from the training set into sequences of actions (uniquely).
- Train NN to predict correct parser action in context using MLE.
- At test time, simply predict actions.
- This gives an f-score of 91.2 on the standard WSJ split. Pretty good - can we do better?

Generative RNNGs

- Traditional distinction of parsing models:
  - *Discriminative* models represent prob dist P(t | w).
  - *Generative* models represent prob dist P(t, w).

- PCFGs are generative models and can be parsed efficiently. More complex generative models can't.

- Generative RNNGs: Keep NT and REDUCE actions, replace SHIFT with GEN(*word*).
  - Get rid of buffer: we are generating the string, not reading it.

Generative RNNGs

	Stack	Terminals	Action
0			NT(S)
1	(S		NT(NP)
2	(S | (NP		GEN(*The*)
3	(S | (NP | *The*	*The*	GEN(*hungry*)
4	(S | (NP | *The* | *hungry*	*The* | *hungry*	GEN(*cat*)
5	(S | (NP | *The* | *hungry* | *cat*	*The* | *hungry* | *cat*	REDUCE
6	(S | (NP *The hungry cat*)	*The* | *hungry* | *cat*	NT(VP)
7	(S | (NP *The hungry cat*) | (VP	*The* | *hungry* | *cat*	GEN(*meows*)
8	(S | (NP *The hungry cat*) | (VP *meows*	*The* | *hungry* | *cat* | *meows*	REDUCE
9	(S | (NP *The hungry cat*) | (VP *meows*)	*The* | *hungry* | *cat* | *meows*	GEN(.)
10	(S | (NP *The hungry cat*) | (VP *meows*) | .	*The* | *hungry* | *cat* | *meows* | .	REDUCE
11	(S (NP *The hungry cat*) (VP *meows*) .)	*The* | *hungry* | *cat* | *meows* | .	

Parsing with generative RNNGs

- We are looking for

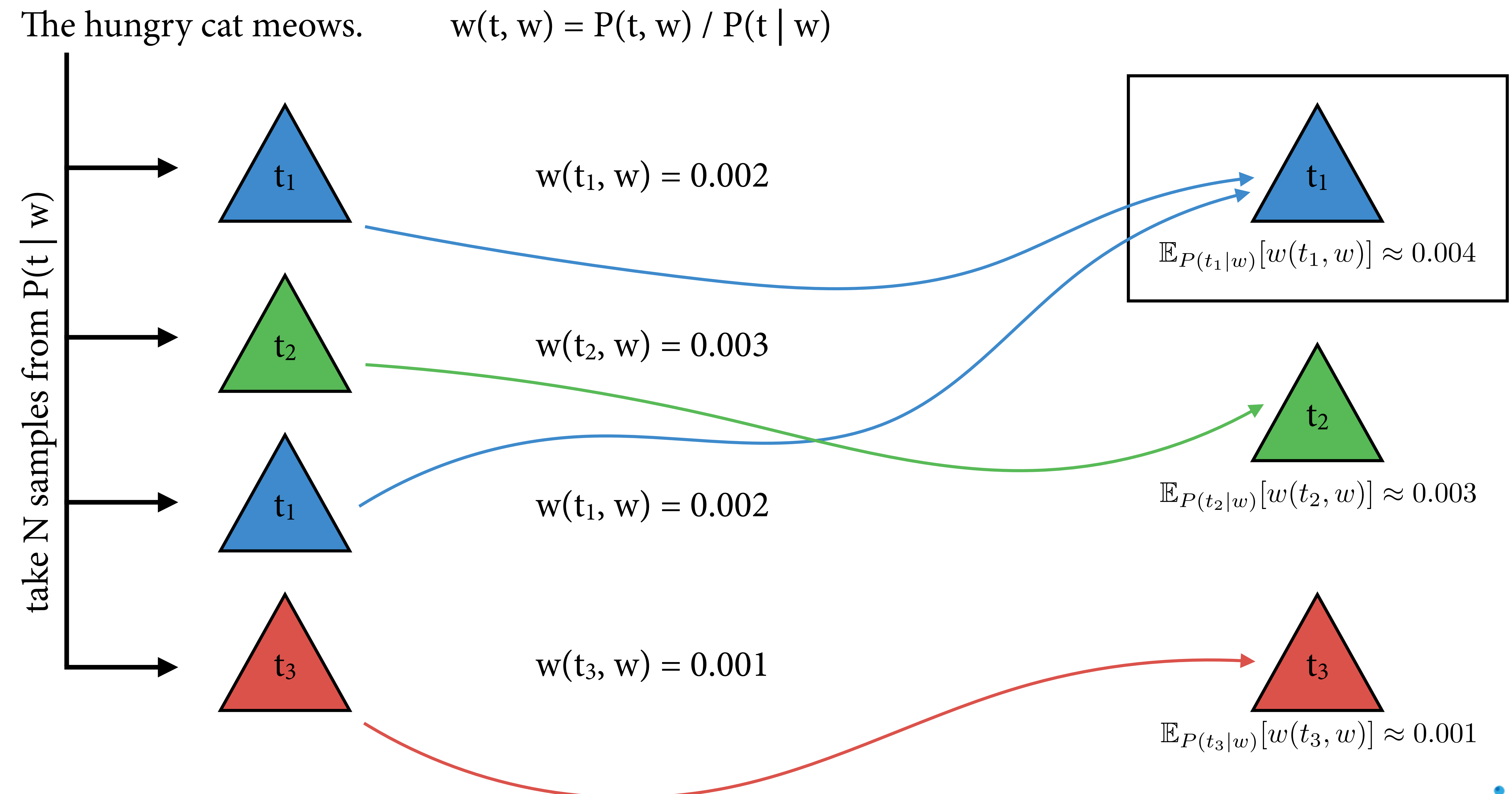
$$\operatorname*{arg\,max}_{t \in \mathsf{parses}(w)} P(t, w),$$

  but the generative RNNG doesn't know anything about w, so it is totally unclear how to compute it.

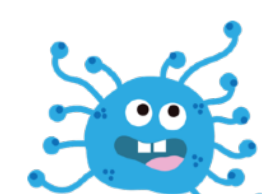
- We *could* get the argmax for P(t | w), but this is not the right model.

- We can also sample from P(t | w). Let's do it many times and reweight → *importance sampling*.

Importance sampling



f-score on PTB: 93.3

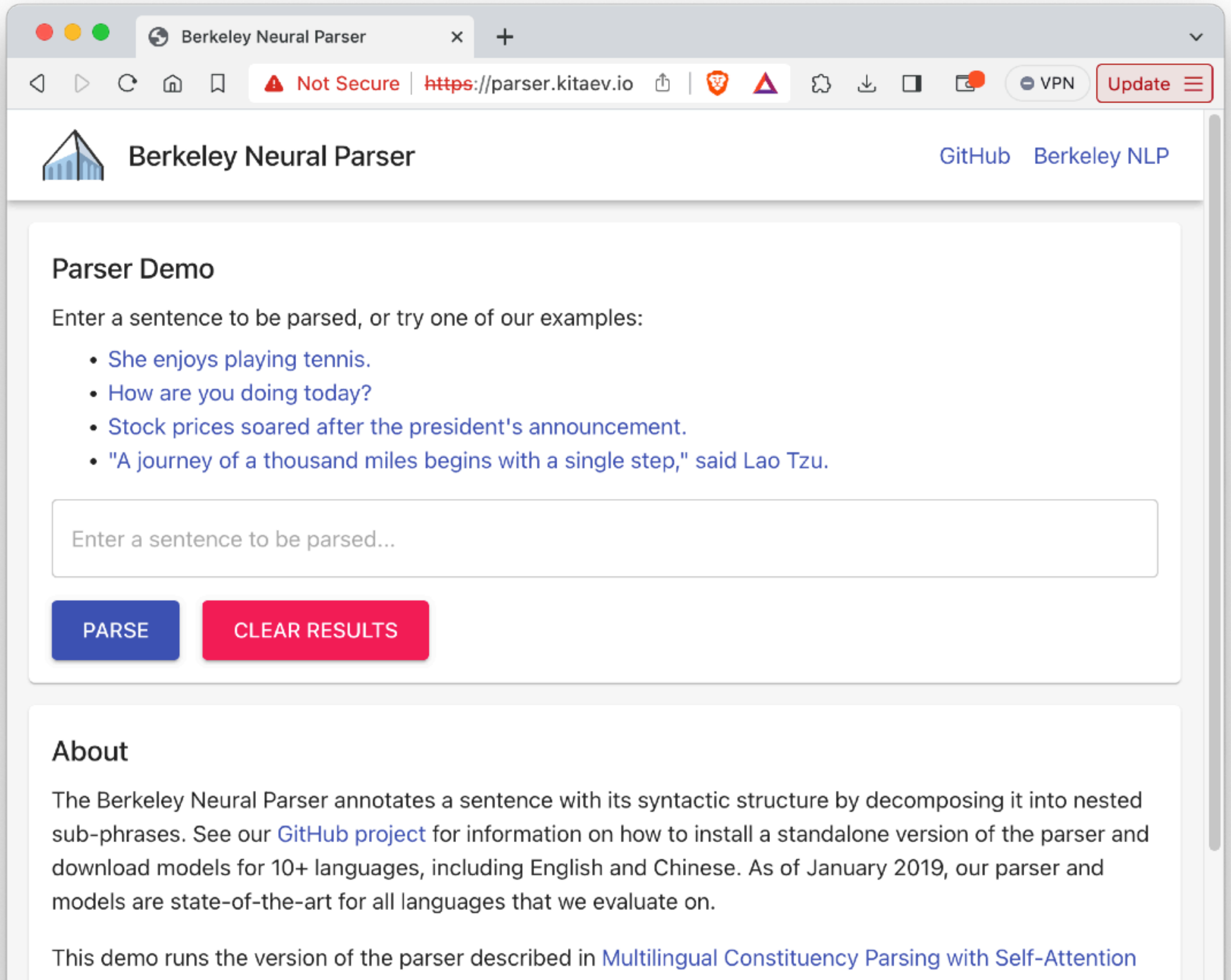


2/3

Neural Berkeley Parser

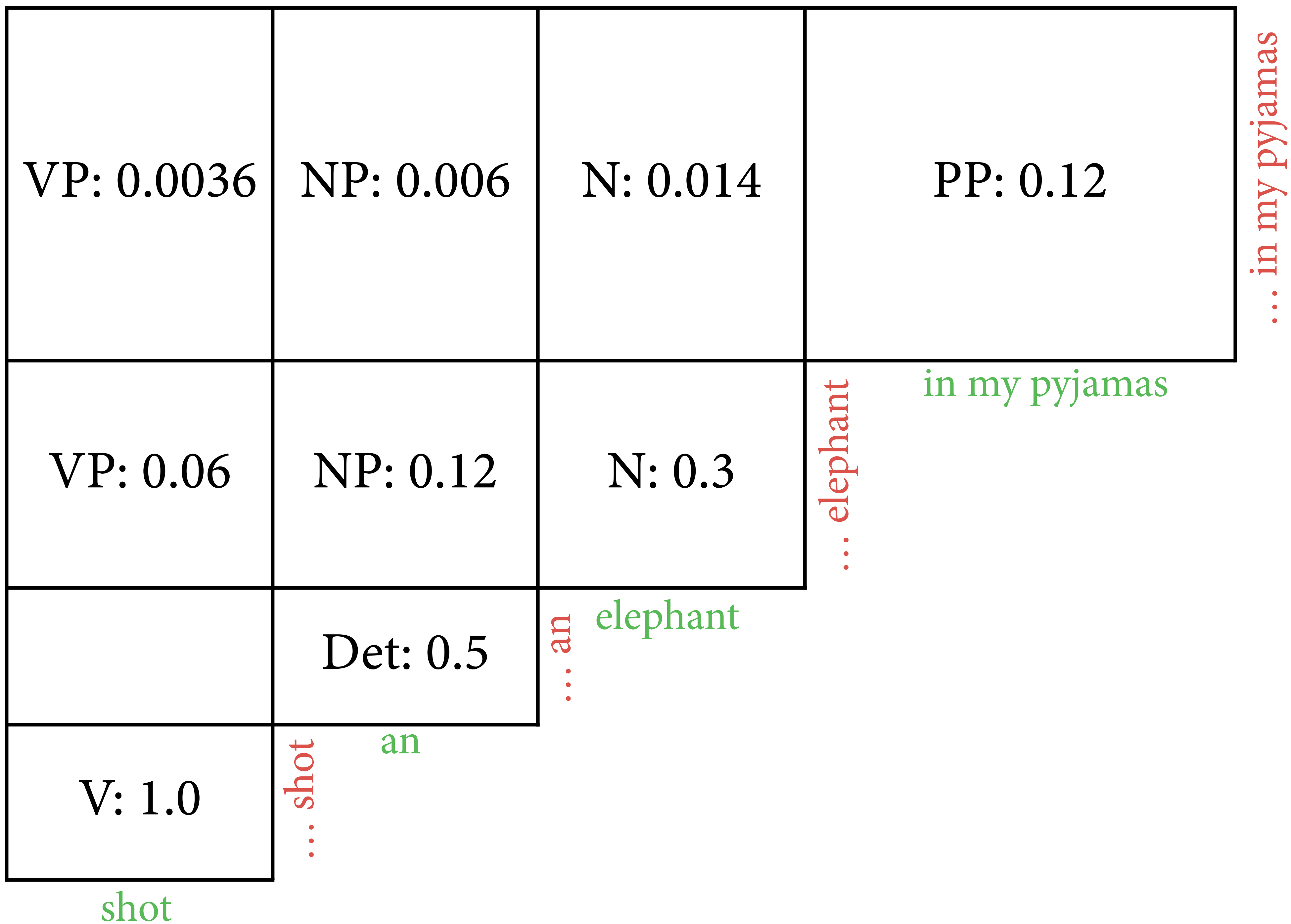
- RNNGs are cool, but they seem complicated.
  - Parallel training is a nightmare.

- Couldn't we just do CKY parsing with neural scores?



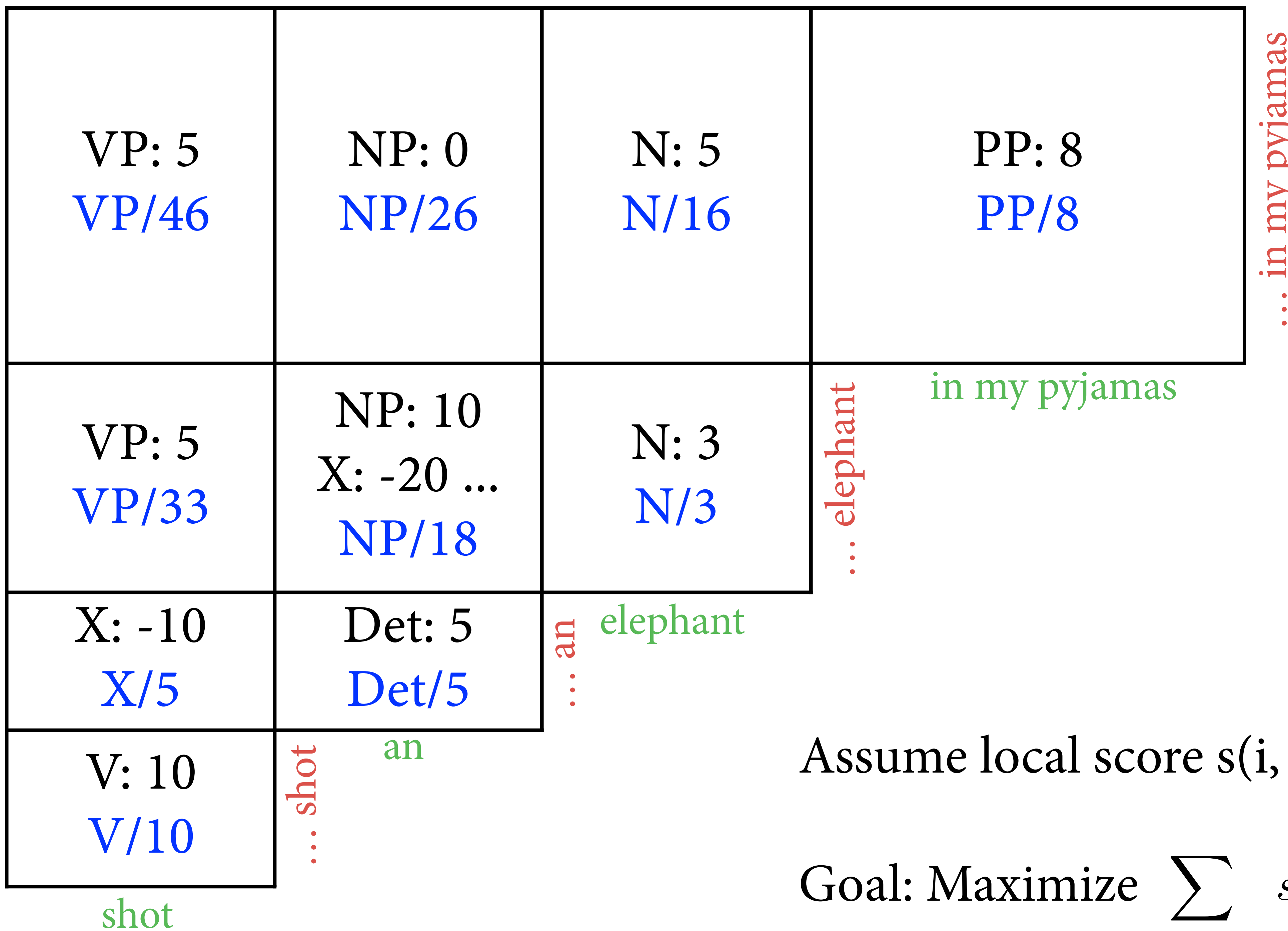
Core of Viterbi-CKY

Viterbi CKY parse chart: $\mathrm{Ch}(i,k) = \{(A,p) \mid p = \max_{d:A \Rightarrow^* w_i \ldots w_{k-1}} P(d)\}$



Core of Viterbi-CKY

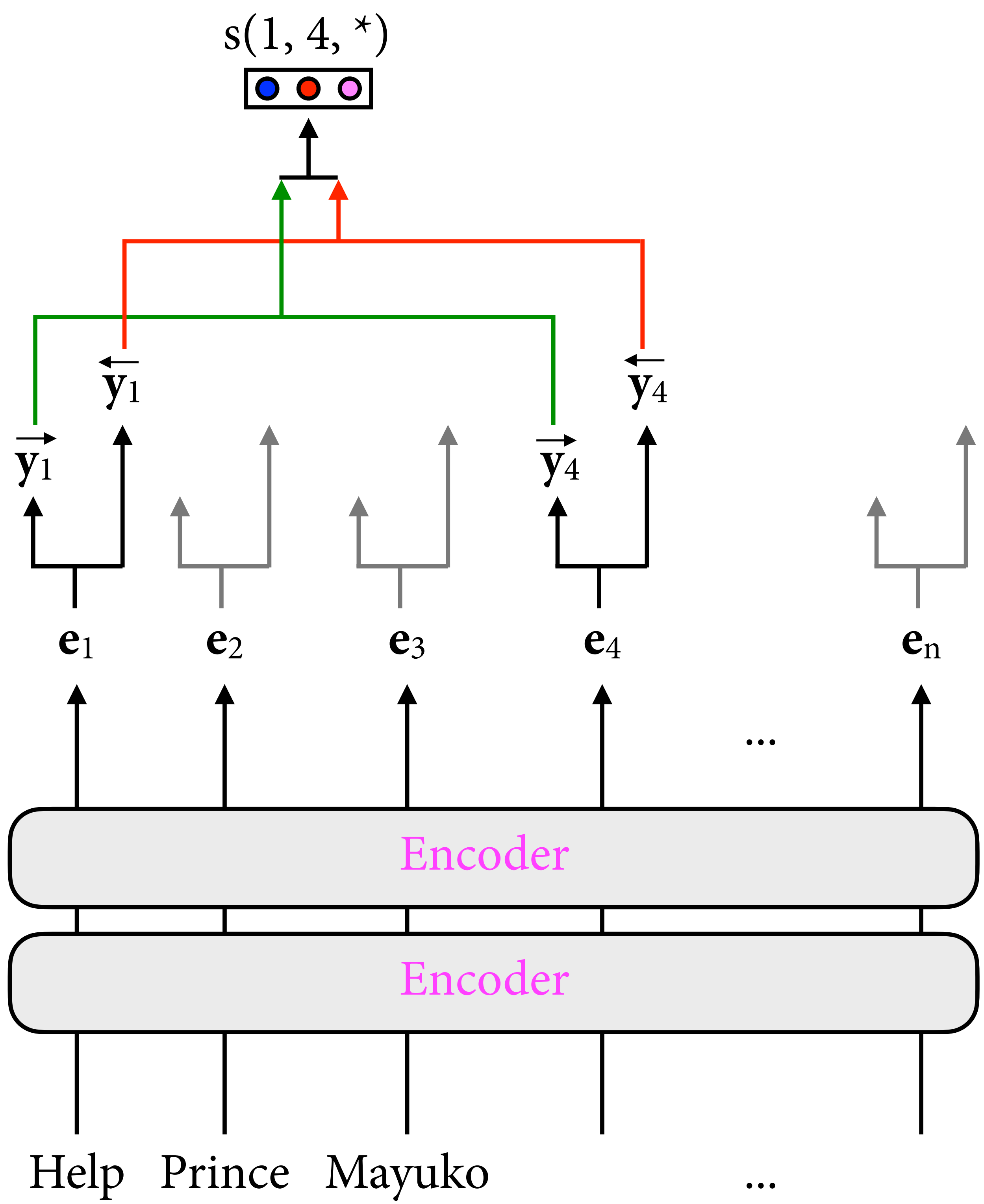
$$\underline{V(i,j)} = \max_{t \text{ for } (i,j)} \sum_{(i',j',\ell)\in t} s(i',j',\ell) = \max_{\ell} s(i,j,\ell) + \max_{k}(V(i,k) + V(k,j))$$



Assume local score s(i, j, l).

Goal: Maximize $\sum_{(i,j,\ell)\in t} s(i,j,\ell)$

Computing local scores



Training uses *hinge loss:* maximize score difference of gold tree to the others.

Evaluation

Encoder Architecture	F1 (dev)	Δ
LSTM (Gaddy et al., 2018)	92.24	-0.43
Self-attentive (Section 2)	92.67	0.00
+ Factored (Section 3)	93.15	0.48
+ CharLSTM (Section 5.1)	93.61	0.94
+ ELMo (Section 5.2)	95.21	2.54

Summary

- Neural models are good at parsing, but neurosymbolic models are better.
  - RNNGs = shift-reduce plus more complex parsing
  - Berkeley = Viterbi-CKY plus more complex training
- Neurosymbolic models are really interesting, but need clearer understanding of design principles.