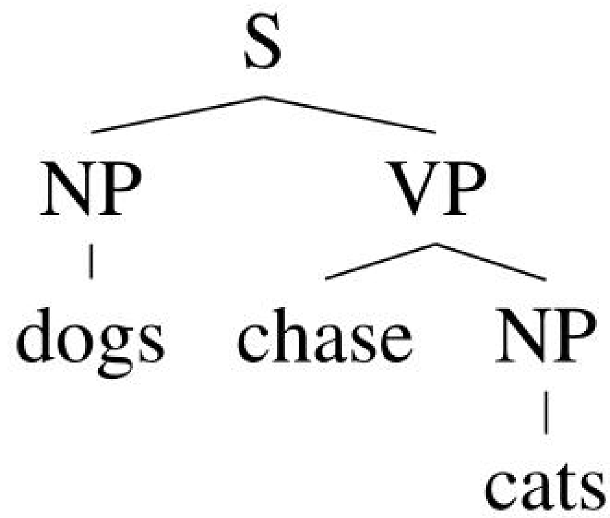
Constituent Parsing as Sequence Labeling

WeiYang ❤️ 51184506043
weiyang@godweiyang.com
https://godweiyang.com
2019.03.28

Parenthesized String

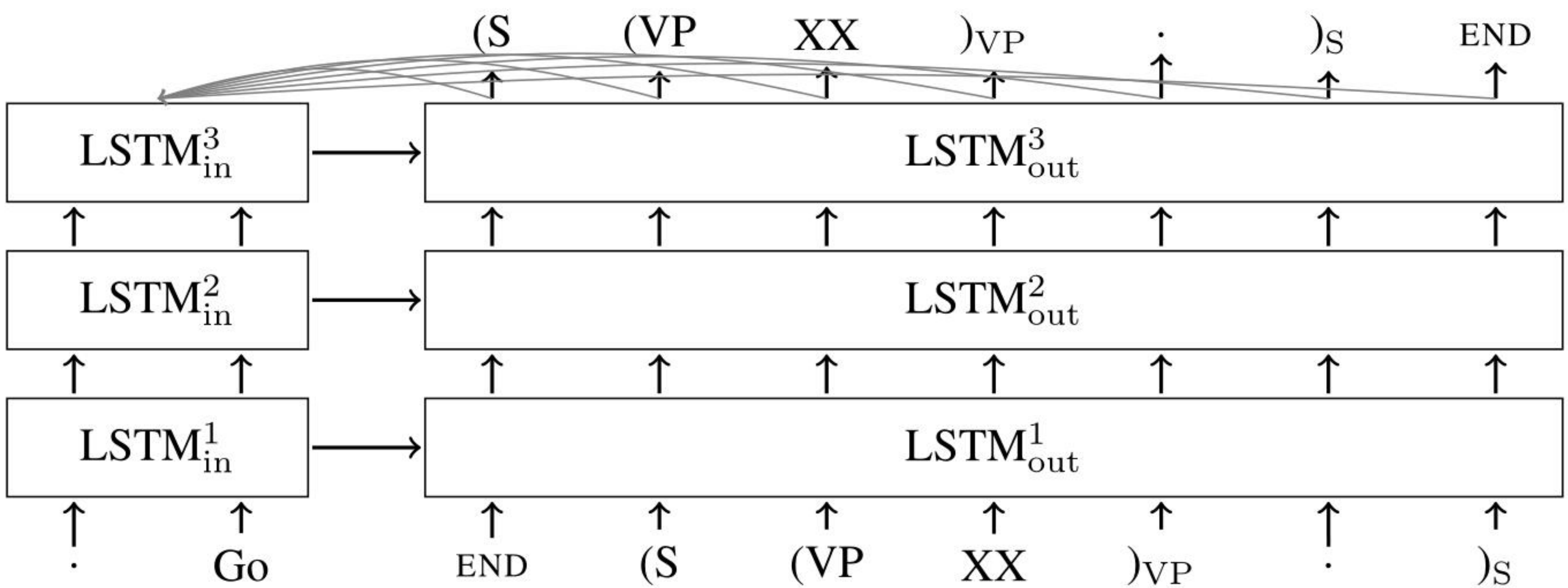


(b)

(S (NP dogs $)_{NP}$ (VP chase (NP cats $)_{NP}$ $)_{VP}$ $)_{S}$

[Choe and Charniak, 2016] Parsing as Language Modeling

Seq2Seq



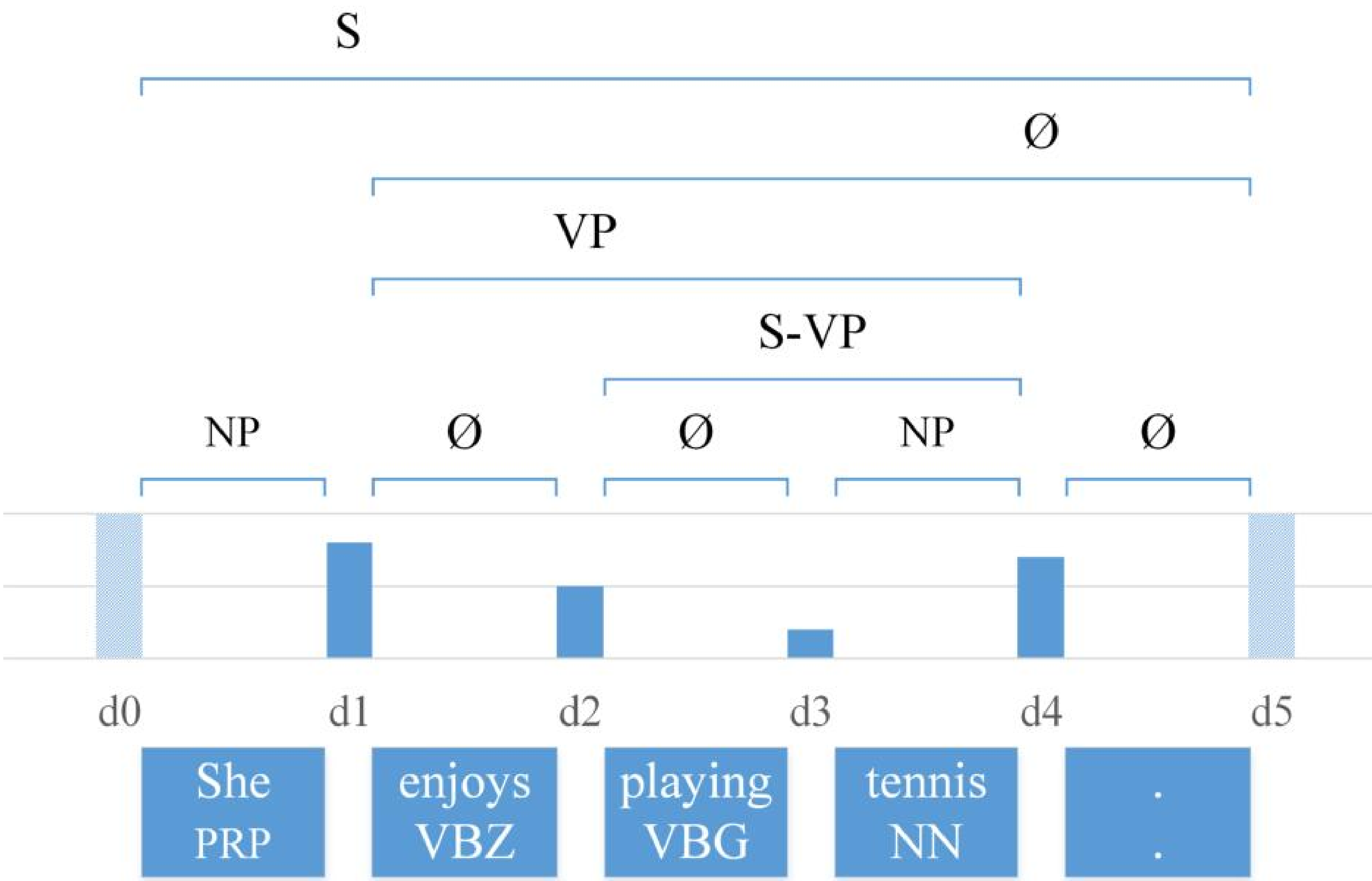
[Vinyals et al., 2015] Grammar as a Foreign Language

Language Model

$$
\begin{aligned}
P(x, y) &= P(z) \\
&= \prod_{t=1}^{m} P(z_t | z_1, \ldots, z_{t-1}) \\
&= \prod_{t=1}^{m} P(z_t | h_t) \\
&= \prod_{t=1}^{m} softmax(W h_t)[z_t]
\end{aligned}
$$

[Choe and Charniak, 2016] Parsing as Language Modeling

Syntactic Distance



[Shen et al., 2018] Straight to the Tree: Constituency Parsing with Neural Syntactic Distance

Parsing as Sequence Labeling

Input:

$$w = [w_1, w_2, \ldots, w_N]$$

Encode:

$$\Phi_N : T_N \to L^{N-1}$$

Model:

$$F_{N,\theta} : V^N \to L^{N-1}$$

Decode:

$$F_{N,\theta} \circ \Phi_N^{-1}$$

[GÃ³mez-RodrÃguez and Vilares, 2018] Constituent Parsing as Sequence Labeling

Encode

Definition:

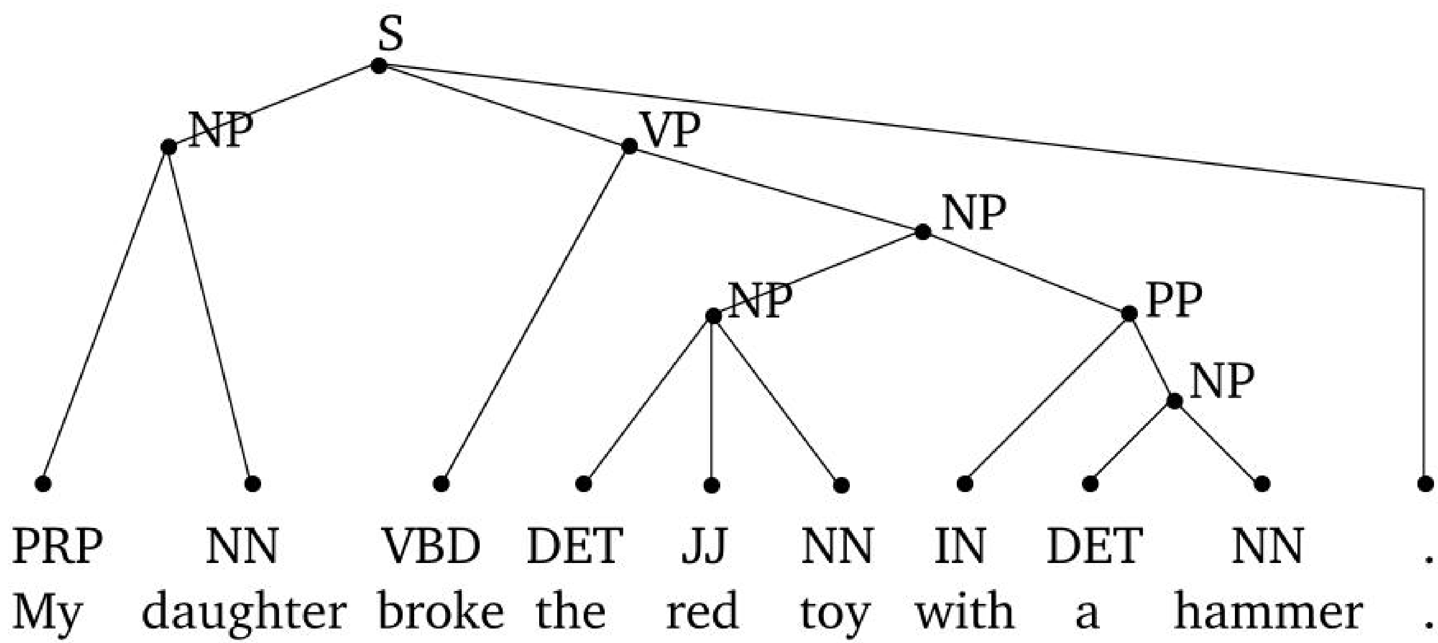
$$\Phi_N : T_N \rightarrow \{(n_i, c_i) | i = 1, 2, \ldots, N-1\}$$

- n_i: number of common ancestors (CA) between w_i and w_{i+1}.
- c_i: label of their lowest common ancestor (LCA).

Properties of Φ_N:

- Complete.
- Injective.
- *Not* surjective.

Example



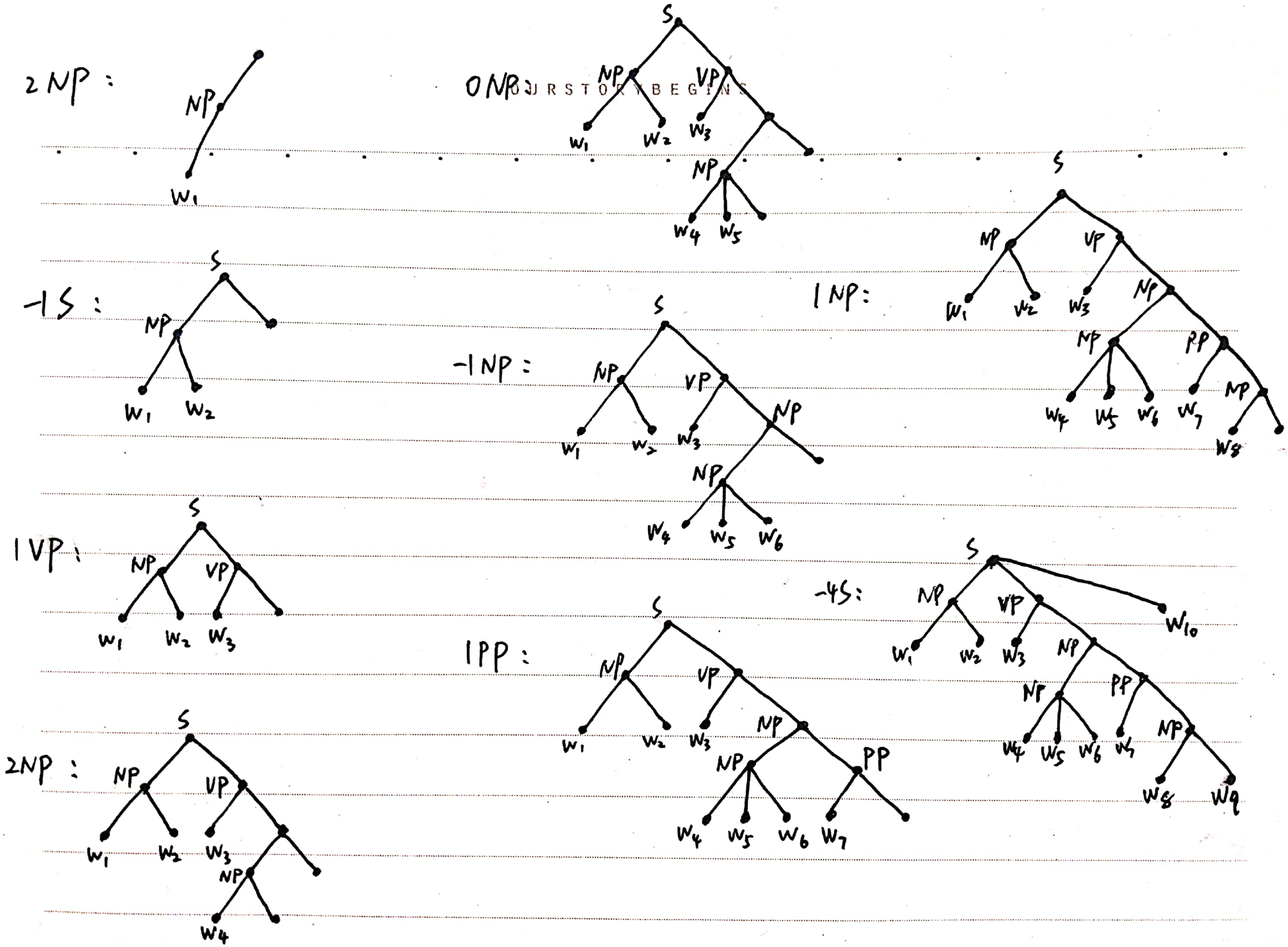
Linearized tree (absolute scale):

2NP 1S 2VP 4NP 4NP 3NP 4PP 5NP 1S

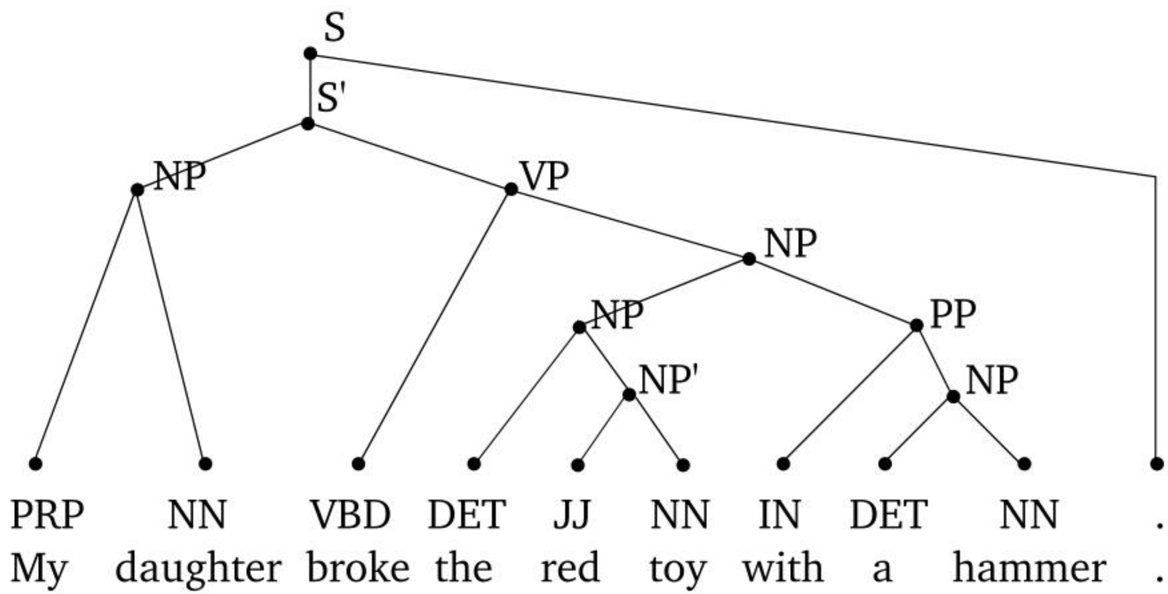
Linearized tree (relative scale):

2NP -1S 1VP 2NP 0NP -1NP 1PP 1NP -4S

Decoding Process



Encoding for k-ary Trees



Linearized tree (absolute scale):

3NP 2S' 3VP 5NP 6NP' 4NP 5PP 6NP 1S

Linearized tree (relative scale):

3NP -1S' 1VP 2NP 1NP' -2NP 1PP 1NP -5S

Linearized tree (simplified relative scale):

3NP -S' 1VP 2NP 1NP' -NP 1PP 1NP -S

Proof

Completeness: Obvious. 😂

Injectivity:

- Parenthesized string:
  $\alpha_0(\bullet_1)\alpha_1(\bullet_2)\ldots\alpha_{|w|-1}(\bullet_{|w|})\alpha_{|w|}$
- Each α_i must be $[)]^*[(X]^*$. So letting $\alpha_i = \alpha_{i)}\alpha_{i(}$, we can get
  $\alpha_{0)}\alpha_{0(}(\bullet_1)\alpha_{1)}\alpha_{1(}(\bullet_2)\ldots(\bullet_{|w|})\alpha_{|w|)}\alpha_{|w|(}$
- Letting $\beta_i = \alpha_{i-1(}(\bullet_i)\alpha_{i)}$, we can get
  $\beta_1\beta_2\ldots\beta_{|w|}$
- β_i contains only one of $[(X]^*(\bullet_i)$ and $(\bullet_i)[)]^*$
- Assigning $\delta(\beta_i)$ to every β_i, we can get
  $\delta(\beta_1)\delta(\beta_2)\ldots\delta(\beta_{|w|-1})$

Limitations

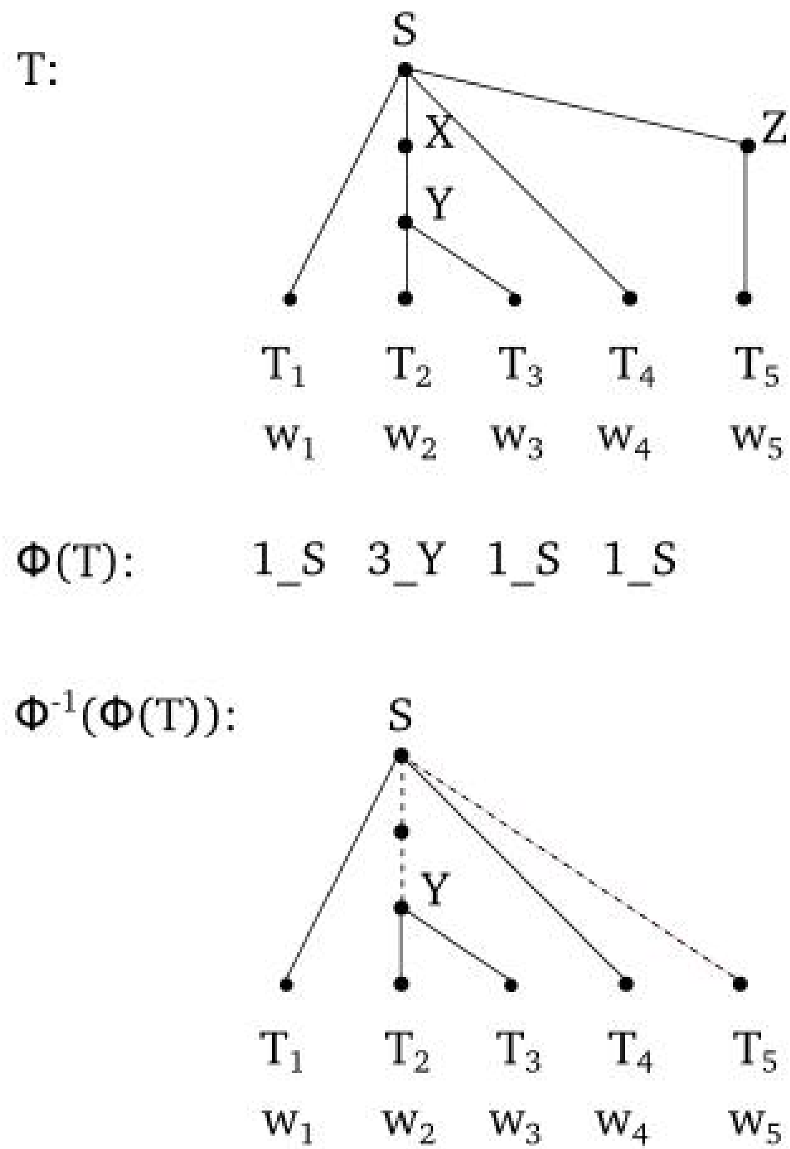
Unary branches:

- Nonterminal nodes:
  $X \to Y \triangleq [X, Y]$
- Leaf nodes:
  - $\Psi_{|w|} : V^{|w|} \to U^{|w|}$
  - (n_i, c_i, u_i)

Non-surjectivity:

- Sequences with conflicting nonterminals.
- Sequences that produce unary structures.

A Example of Non-surjectivity



Experiments

Model	Testbed	CPU Run #Cores	CPU Run Sents/s	GPU Run #GPU	GPU Run Sents/s	F-score
Sequence labeling						
$\text{MLP}_e^{\Psi,\Phi}$	WSJ23	1	501	1	669	74.1
$\text{MLP}_e^{\Phi'}$	WSJ23	1	349	1	929	74.8
$\text{BILSTM}_{m=1,e}^{\Psi,\Phi}$	WSJ23	1	148	1	581	88.1
$\text{BILSTM}_{m=1,e}^{\Phi'}$	WSJ23	1	221	1	1016	88.3
$\text{BILSTM}_{m=2,e,ch}^{\Psi,\Phi}$	WSJ23	1	66	1	434	89.9
$\text{BILSTM}_{m=2,e,ch}^{\Phi'}$	WSJ23	1	115	1	780	90.0
$\text{BILSTM}_{m=2,e}^{\Psi,\Phi}$	WSJ23	1	74	1	506	90.0
$\text{BILSTM}_{m=2,e}^{\Phi'}$	WSJ23	1	126	1	898	90.0
Sequence-to-sequence						
3-layer LSTM	WSJ 23					<70
3-layer LSTM + Attention$^\diamond$ (Vinyals et al., 2015)	WSJ 23	Multi-core (number not specified)	120			88.3
Constituency parsing as dependency parsing						
Fernández-González and Martins (2015)$^\diamond$	WSJ23	1	41			90.2
Chart-based parsers						
Charniak (2000)*	WSJ23	1	6			89.5
Petrov and Klein (2007)*	WSJ23	1	6			90.1
Stern et al. (2017)$^\diamond$	WSJ23	16*	20			91.8
Kitaev and Klein (2018) +ELMo (Peters et al., 2018)$^\diamond$	WSJ23			2	70	95.1
Chart-based parsers with GPU-specific implementation						
Canny et al. (2013)$^\diamond$	WSJ(<30)			1	250	
Hall et al. (2014)$^\diamond$	WSJ(<40)			1	404	
Transition-based and other greedy constituent parsers						
Zhu et al. (2013)$^\diamond$	WSJ23	1	101			89.9
Zhu et al. (2013)+Padding$^\diamond$	WSJ23	1	90			90.4
Dyer et al. (2016)$^\triangleright$	WSJ23	1	17			91.2
Fernández and Gómez-Rodríguez (2018)$^\diamond$	WSJ23	1	18			91.7
Stern et al. (2017)$^\diamond$	WSJ23	16*	76			91.8
Liu and Zhang (2017)	WSJ23					91.8
Shen et al. (2018)	WSJ23			1	111	91.8

Comparison with Syntactic Distance

- Must be binary trees.
- Top-down recursion.
- Lack theoretical properties which can cause ambiguity (e.g., a sequence of $n-1$ equal labels in this encoding can represent any binary tree with n leaves).

[GÃ³mez-RodrÃguez and Vilares, 2018] Constituent Parsing as Sequence Labeling

Other Limitations

High error rate on closing brackets:

- Dynamic encodings.

Sparsity:

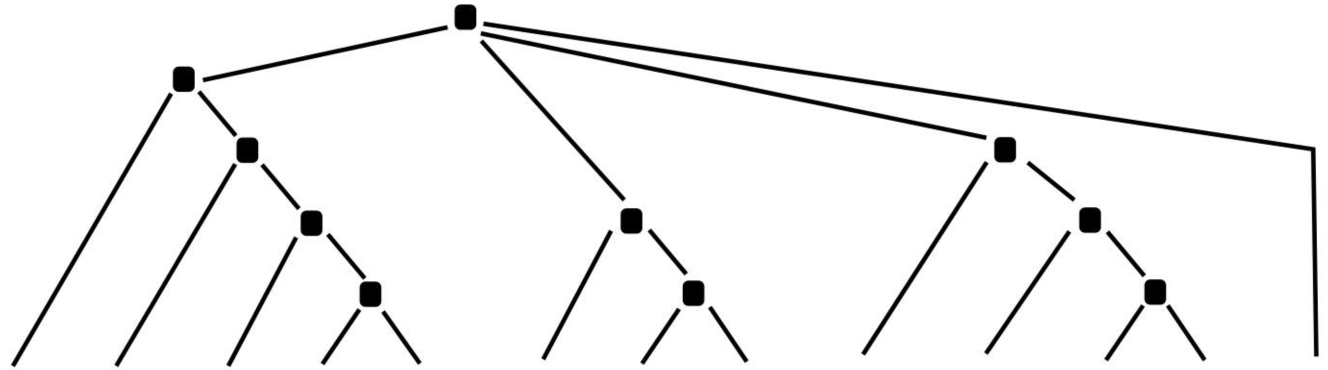
- Decomposition of the label space.

Greedy decoding:

- Auxiliary tasks.
- Policy gradient fine-tuning.

[Vilares et al., 2019] Better, Faster, Stronger Sequence Tagging Constituent Parsers

Dynamic Encodings



	a	b	c	d	e	f	g	h	i	j	k	l	m
Relative:	2	1	1	1	-4	1	1	-2	1	1	1	-3	∅
Absolute:	2	3	4	5	1	2	3	1	2	3	4	1	∅
Dynamic:	2_r	1_r	1_r	1_r	1_a	1_r	1_r	1_a	1_r	1_r	1_r	1_a	∅

Replace Φ_N with Ω_N iff:

- $\Omega_N(w_t) = (n'_t, c'_t, u'_t)$ with $n'_t \leq 3$.
- $\Phi_N(w_t) = (n_t, c_t, u_t)$ with $n_t \leq -2$.

[Vilares et al., 2019] Better, Faster, Stronger Sequence Tagging Constituent Parsers

Decomposition of the Label Space

Method:

- $(n_t, c_t, u_t) \in L \to n_t \in N, c_t \in C, u_t \in U$
- $|N| \times |C| \times |U| \to |N| + |C| + |U|$

Objective:

- $\mathcal{L} = \mathcal{L}_n + \mathcal{L}_c + \mathcal{L}_u$

Hard-sharing architecture:

- task$_U$ $\to$ task$_N$ and/or task$_C$, etc.
- No improvement. 😤

[Vilares et al., 2019] Better, Faster, Stronger Sequence Tagging Constituent Parsers

Auxiliary Tasks

Partial labels:

- Predict $n_t, \ldots, n_{t+k}$ at time step t.
- Usually set $|k| = 1$.

Syntactic distances:

- Provide different types of contextual information.

Objective:

- $\mathcal{L} = \mathcal{L}_n + \mathcal{L}_c + \mathcal{L}_u + \beta \sum_a \mathcal{L}_a$

Policy Gradient Fine-tuning

Original:

- $\Delta_\theta \log \pi(l_t | s_t; \theta) R_{tree}$

My viewpoint: 😳

- $p(tree; \theta) = \prod_{t=1}^{N-1} \pi(l_t | s_t; \theta)$
- $\nabla_\theta L(\theta) = \sum_{tree} R_{tree} \nabla_\theta p(tree; \theta)$

$$= \sum_{tree} p(tree; \theta) R_{tree} \nabla_\theta \log p(tree; \theta)$$

$$\approx \sum_{tree \in T} R_{tree} \nabla_\theta \log p(tree; \theta)$$

[Vilares et al., 2019] Better, Faster, Stronger Sequence Tagging Constituent Parsers

Policy Gradient Fine-tuning

Variance reduction:

- $\Delta_\theta(\log \pi(l_t|s_t;\theta) + N)(R_{tree} - B_{tree})$
  $+\beta\Delta_\theta H(\pi(s_t;\theta) + N)$

My viewpoint: ❓

- $\nabla_\theta(\log p(tree;\theta) + N)(R_{tree} - B_{tree})$
  $+\beta\Delta_\theta H(p(tree;\theta) + N)$

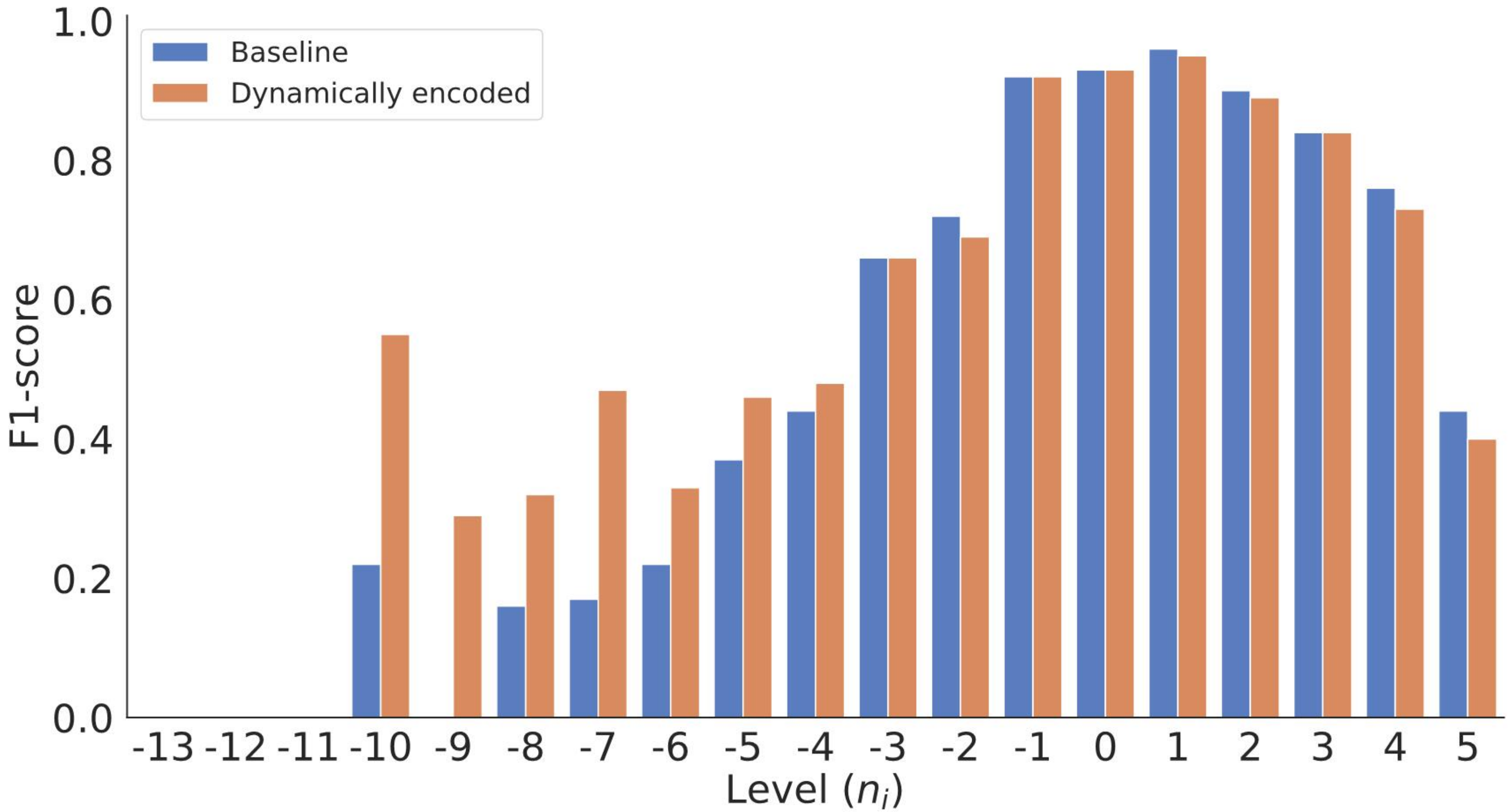
[Vilares et al., 2019] Better, Faster, Stronger Sequence Tagging Constituent Parsers

Experiment

Model	**F-score**	**(+/-)**	**Sents/s**
Gómez and Vilares (2018)	89.70	-	109
Our baseline	89.77	(+0.07)	111
+ DE	90.22	(+0.52)	111
+ MTL	90.38	(+0.68)	130
aux(n_{t+1})	90.41	(+0.71)	130
aux(n_{t-1})	90.57	(+0.87)	130
aux(distances)	90.55	(+0.85)	130
+ PG	**90.70**	(+1.00)	130

[Vilares et al., 2019] Better, Faster, Stronger Sequence Tagging Constituent Parsers

Comparison



[Vilares et al., 2019] Better, Faster, Stronger Sequence Tagging Constituent Parsers