Dependency Transformer Grammars:

Integrating Dependency Structures into Transformer Language Models

Yida Zhao, Chao Lou, Kewei Tu
School of Information Science and Technology, ShanghaiTech University

Dependency Transformer Grammars (DTGs)

- Syntactic language models that jointly model parse trees and strings
  - Autoregressively generate a transition/action sequence

- DTGs model transition sequences of transition-based dependency parsers
  - Use Arc-standard transition systems
  - Replace each **SHIFT** in Arc-standard with generating a new token

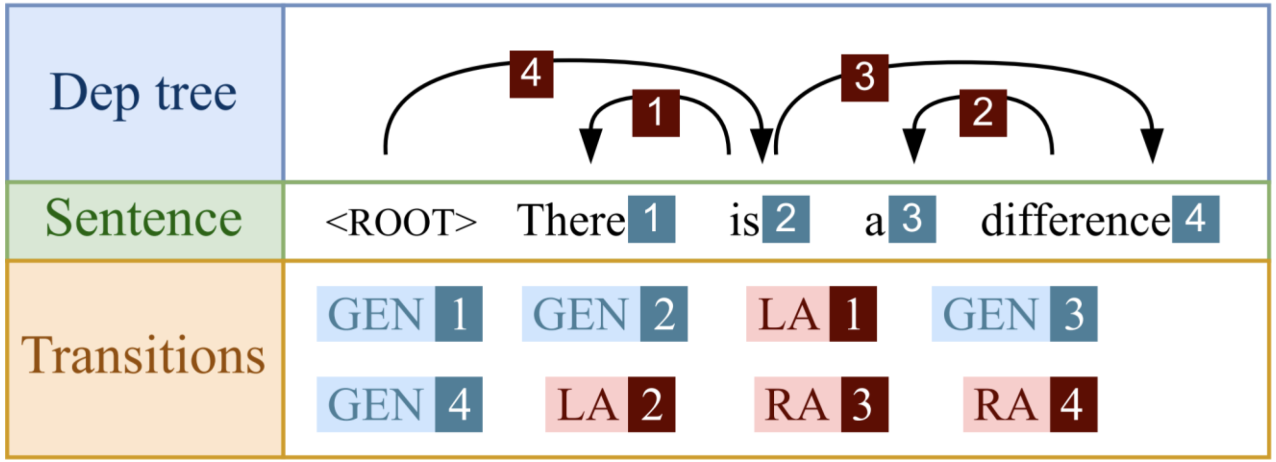
Arc-Standard Transition Systems

arc-standard

Shift $(\sigma, i|\beta, A) \Rightarrow (\sigma|i, \beta, A)$

LArc $(\sigma|i|j, \beta, A) \Rightarrow (\sigma|j, \beta, A \cup \{(j \rightarrow i)\})$

RArc $(\sigma|i|j, \beta, A) \Rightarrow (\sigma|i, \beta, A \cup \{(i \rightarrow j)\})$



Dep tree

Sentence

<ROOT> There 1 is 2 a 3 difference 4

Transitions

GEN 1 GEN 2 LA 1 GEN 3

GEN 4 LA 2 RA 3 RA 4

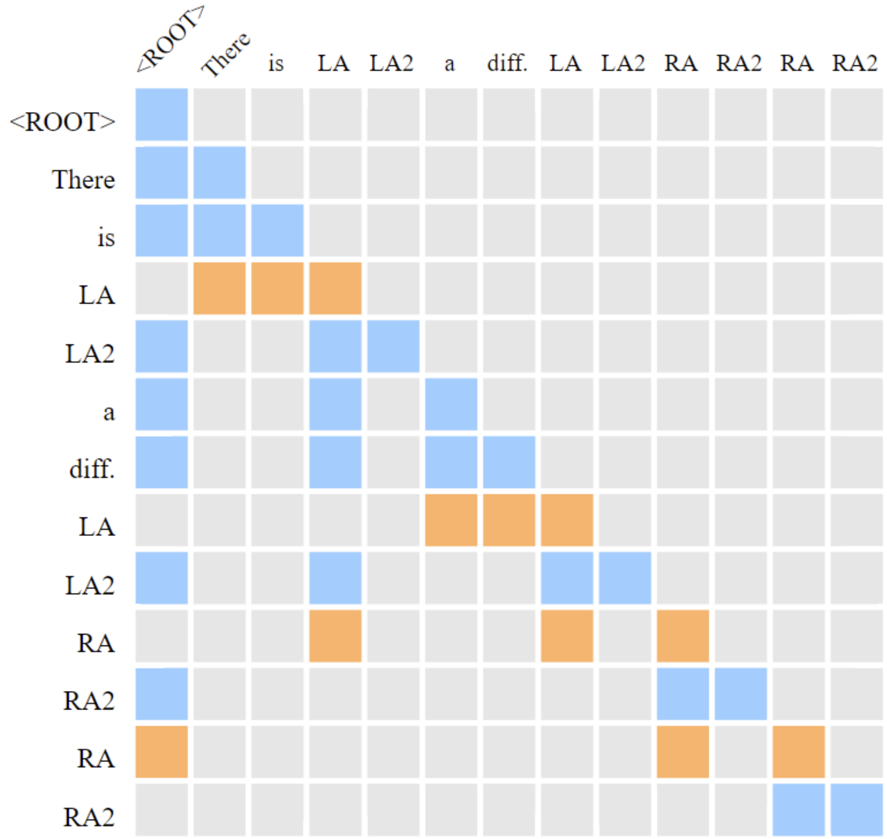
Constrained Attention Patterns

- Design two constrained attention patterns to simulate the stack in the parsing system
  - **STACK** attention for gathering information in the stack and predicting a new transition
  - **COMPOSE** attention for composing the information from a head-dependent pair and replace them with a composition in the stack

Attention Masks

- Construct attention masks for each pattern that force the stack information gathering and head-dependent representation learning
  - Duplicate the arc transition to perform both **COMPOSE** and **STACK**

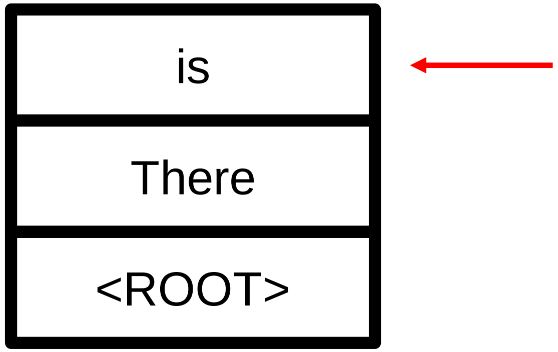
i	Input	Attn. Mask	Prediction
0	<ROOT>	STACK	GEN(There)
1	There	STACK	GEN(is)
2	is	STACK	LEFTARC
3	LEFTARC + is	COMPOSE	-
4	LEFTARC2 + is	STACK	GEN(a)
5	a	STACK	GEN(difference)
6	difference	STACK	LEFTARC
7	LEFTARC + difference	COMPOSE	-
8	LEFTARC2 + difference	STACK	RIGHTARC
9	RIGHTARC + is	COMPOSE	-
10	RIGHTARC2 + is	STACK	RIGHTARC
11	RIGHTARC + <ROOT>	COMPOSE	-
12	RIGHTARC2 + <ROOT>	STACK	<END>

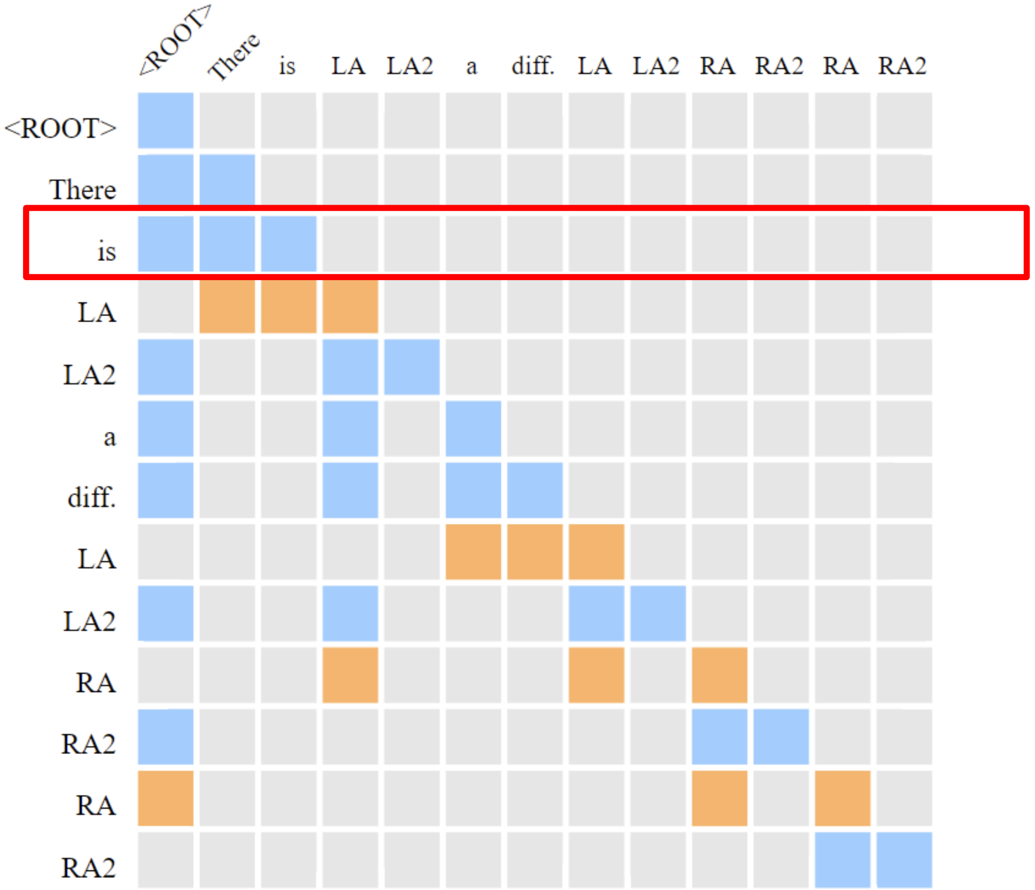


	<ROOT>	There	is	LA	LA2	a	diff.	LA	LA2	RA	RA2	RA	RA2
<ROOT>	🟦	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜
There	🟦	🟦	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜
is	🟦	🟦	🟦	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜
LA	⬜	🟧	🟧	🟧	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜
LA2	🟦	⬜	⬜	🟦	🟦	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜
a	🟦	⬜	⬜	🟦	⬜	🟦	⬜	⬜	⬜	⬜	⬜	⬜	⬜
diff.	🟦	⬜	⬜	🟦	⬜	🟦	🟦	⬜	⬜	⬜	⬜	⬜	⬜
LA	⬜	⬜	⬜	⬜	⬜	🟧	🟧	🟧	⬜	⬜	⬜	⬜	⬜
LA2	🟦	⬜	⬜	🟦	⬜	⬜	⬜	🟦	🟦	⬜	⬜	⬜	⬜
RA	⬜	⬜	⬜	🟧	⬜	⬜	⬜	🟧	⬜	🟧	⬜	⬜	⬜
RA2	🟦	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	🟦	🟦	⬜	⬜
RA	🟧	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	🟧	⬜	🟧	⬜
RA2	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	🟦	🟦

STACK attention

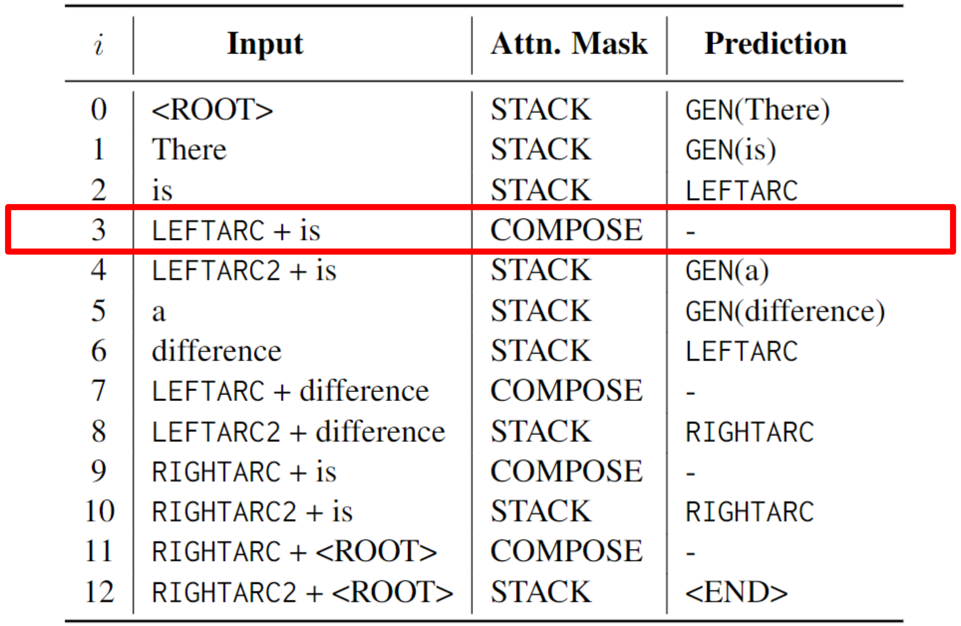
i	**Input**	**Attn. Mask**	**Prediction**
0	<ROOT>	STACK	GEN(There)
1	There	STACK	GEN(is)
2	is	STACK	LEFTARC
3	LEFTARC + is	COMPOSE	-
4	LEFTARC2 + is	STACK	GEN(a)
5	a	STACK	GEN(difference)
6	difference	STACK	LEFTARC
7	LEFTARC + difference	COMPOSE	-
8	LEFTARC2 + difference	STACK	RIGHTARC
9	RIGHTARC + is	COMPOSE	-
10	RIGHTARC2 + is	STACK	RIGHTARC
11	RIGHTARC + <ROOT>	COMPOSE	-
12	RIGHTARC2 + <ROOT>	STACK	<END>



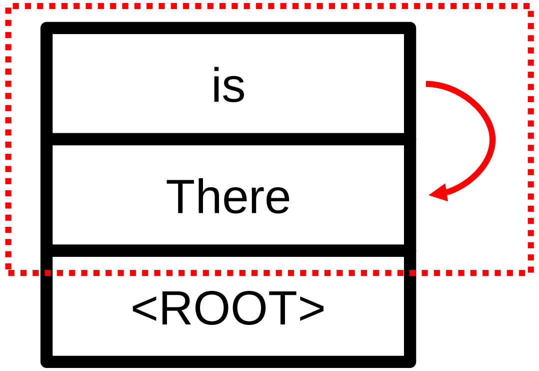


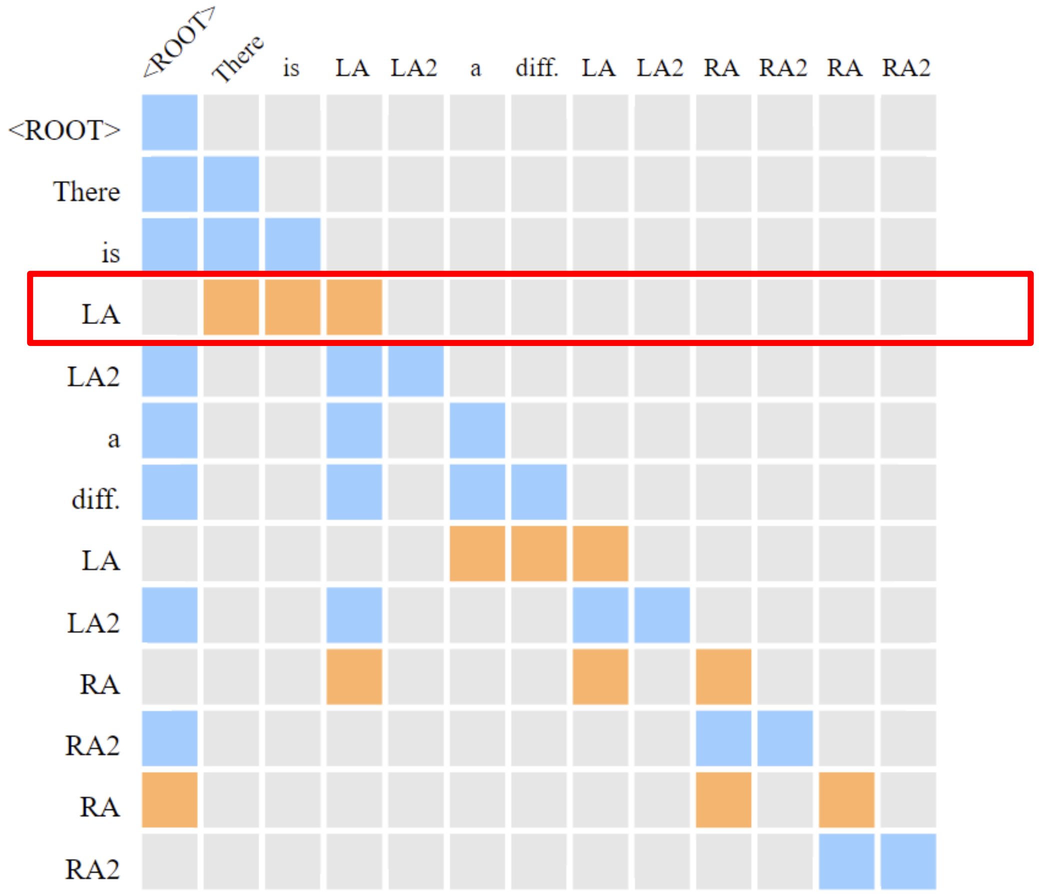
	<ROOT>	There	is	LA	LA2	a	diff.	LA	LA2	RA	RA2	RA	RA2
<ROOT>	🟦												
There	🟦	🟦											
is	🟦	🟦	🟦										
LA		🟧	🟧	🟧									
LA2	🟦			🟦	🟦								
a	🟦			🟦		🟦							
diff.	🟦			🟦		🟦	🟦						
LA						🟧	🟧	🟧					
LA2	🟦			🟦				🟦	🟦				
RA				🟧				🟧		🟧			
RA2	🟦									🟦	🟦		
RA	🟧									🟧		🟧	
RA2												🟦	🟦

COMPOSE attention



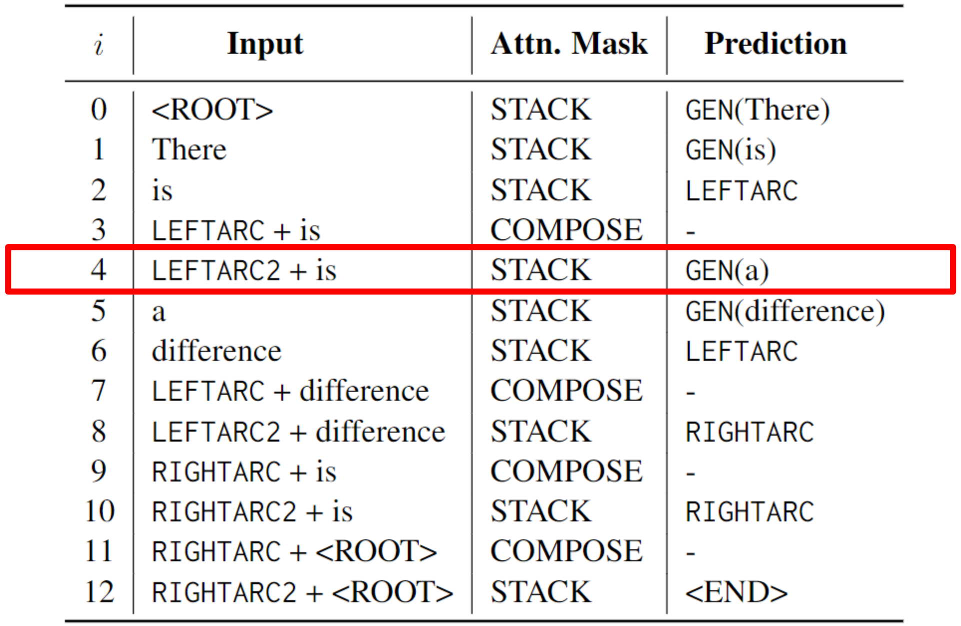
i	Input	Attn. Mask	Prediction
0	<ROOT>	STACK	GEN(There)
1	There	STACK	GEN(is)
2	is	STACK	LEFTARC
3	LEFTARC + is	COMPOSE	-
4	LEFTARC2 + is	STACK	GEN(a)
5	a	STACK	GEN(difference)
6	difference	STACK	LEFTARC
7	LEFTARC + difference	COMPOSE	-
8	LEFTARC2 + difference	STACK	RIGHTARC
9	RIGHTARC + is	COMPOSE	-
10	RIGHTARC2 + is	STACK	RIGHTARC
11	RIGHTARC + <ROOT>	COMPOSE	-
12	RIGHTARC2 + <ROOT>	STACK	<END>



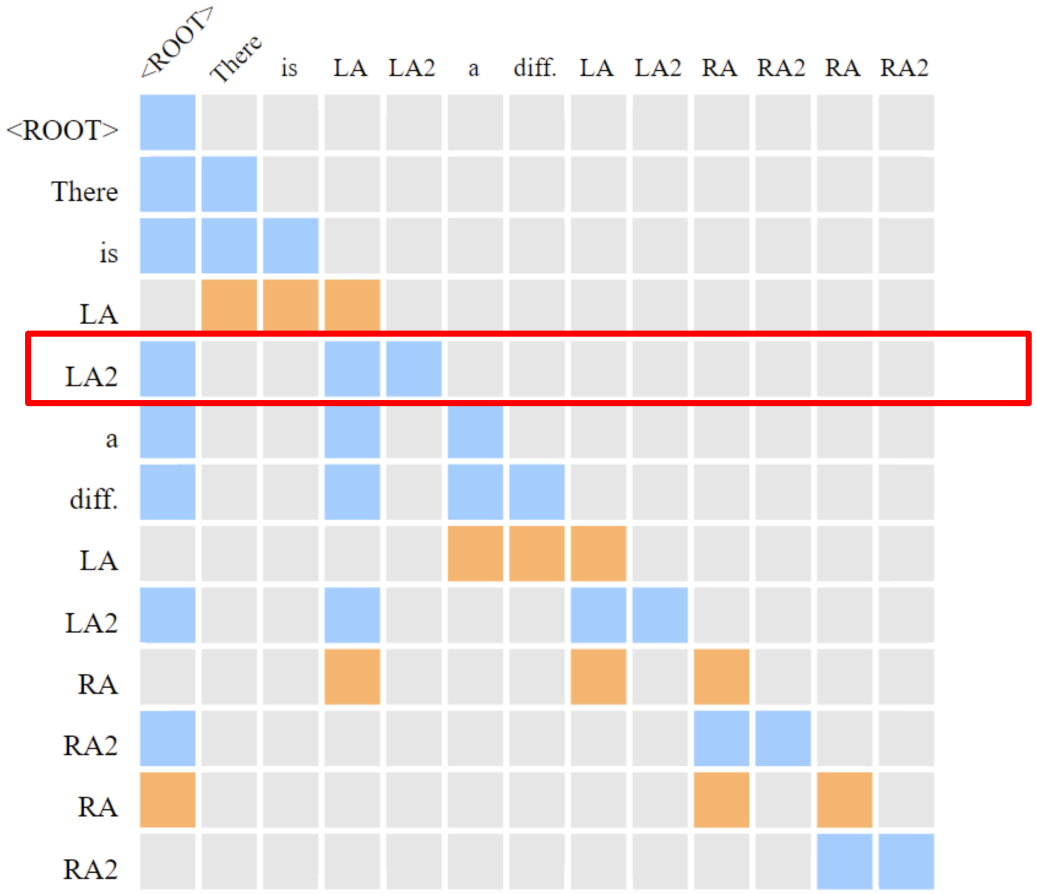


	<ROOT>	There	is	LA	LA2	a	diff.	LA	LA2	RA	RA2	RA	RA2
<ROOT>	🟦												
There	🟦	🟦											
is	🟦	🟦	🟦										
LA		🟧	🟧	🟧									
LA2	🟦			🟦	🟦								
a	🟦			🟦		🟦							
diff.	🟦			🟦		🟦	🟦						
LA						🟧	🟧	🟧					
LA2	🟦			🟦				🟦	🟦				
RA				🟧				🟧		🟧			
RA2	🟦									🟦	🟦		
RA	🟧									🟧		🟧	
RA2												🟦	🟦

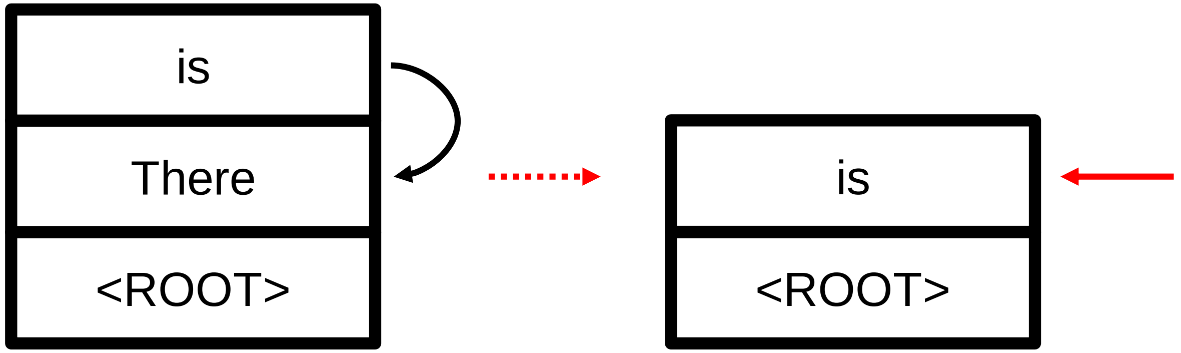
STACK attention



i	Input	Attn. Mask	Prediction
0	<ROOT>	STACK	GEN(There)
1	There	STACK	GEN(is)
2	is	STACK	LEFTARC
3	LEFTARC + is	COMPOSE	-
4	LEFTARC2 + is	STACK	GEN(a)
5	a	STACK	GEN(difference)
6	difference	STACK	LEFTARC
7	LEFTARC + difference	COMPOSE	-
8	LEFTARC2 + difference	STACK	RIGHTARC
9	RIGHTARC + is	COMPOSE	-
10	RIGHTARC2 + is	STACK	RIGHTARC
11	RIGHTARC + <ROOT>	COMPOSE	-
12	RIGHTARC2 + <ROOT>	STACK	<END>



	<ROOT>	There	is	LA	LA2	a	diff.	LA	LA2	RA	RA2	RA	RA2
<ROOT>	🟦												
There	🟦	🟦											
is	🟦	🟦	🟦										
LA		🟧	🟧	🟧									
LA2	🟦			🟦	🟦								
a	🟦			🟦		🟦							
diff.	🟦			🟦		🟦	🟦						
LA						🟧	🟧	🟧					
LA2	🟦			🟦				🟦	🟦				
RA				🟧				🟧		🟧			
RA2	🟦									🟦	🟦		
RA	🟧									🟧		🟧	
RA2												🟦	🟦



Relative Positional Encoding

- Transformer-XL based positional encoding
  - Using the relative depth in the stack for **STACK** attention.

$$R_{ij} = d(i) - d(j)$$

  - Using 0 and -1 for head and dependent for **COMPOSE** attention.

$$R_{ij} \begin{cases} 0 \;\; if\ \boldsymbol{j}\ is\ the\ head \\ -1 \;\; if\ \boldsymbol{j}\ is\ the\ dependent \end{cases}$$

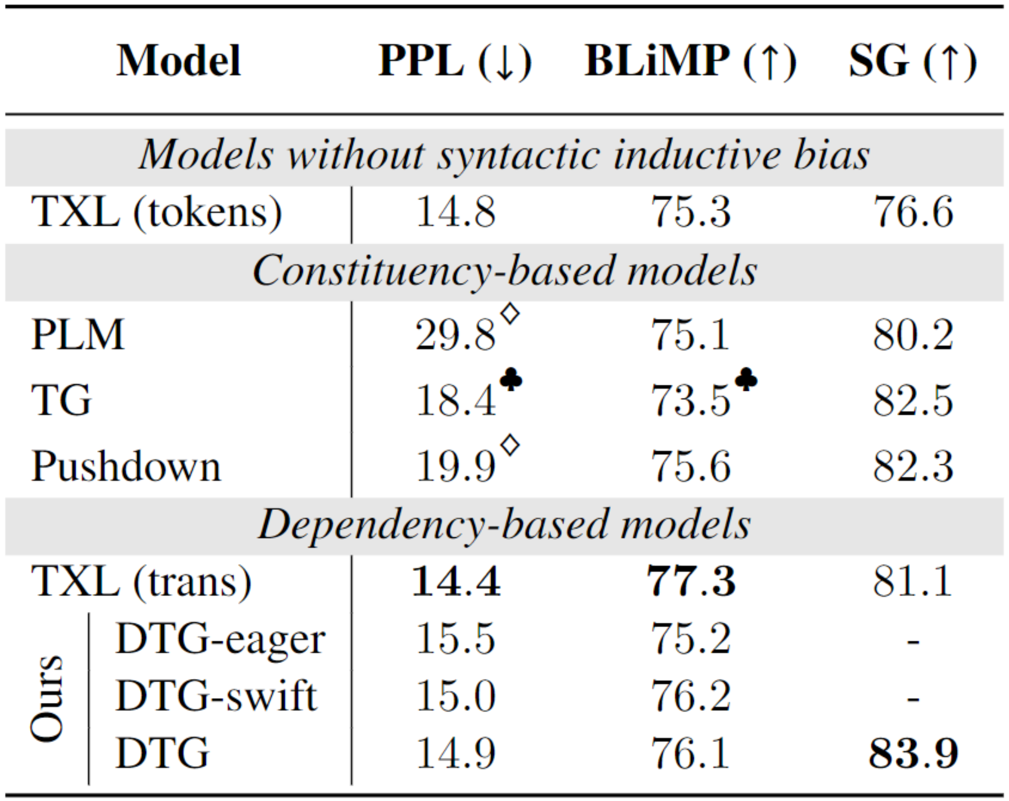
Arc Representation

- Each LA and RA is represented by a combination of the special $LEFTARC/RIGHTARC$ token and the head token

$$E(LA/RA) = E(LEFTARC/RIGHTARC) + E(head\ token)$$

Experiments

- Evaluate sentence-level perplexity and syntactic generalization
  - Compare DTGs with Transformer LM baselines and constituency-based syntactic LMs
  - Compare Arc-standard system with other dependency transition systems for syntactic LM supervision
  - Better syntactic generalization and comparable perplexity !



	Model	PPL (↓)	BLiMP (↑)	SG (↑)
Models without syntactic inductive bias				
	TXL (tokens)	14.8	75.3	76.6
Constituency-based models				
	PLM	29.8$^{\diamond}$	75.1	80.2
	TG	18.4$^{\clubsuit}$	73.5$^{\clubsuit}$	82.5
	Pushdown	19.9$^{\diamond}$	75.6	82.3
Dependency-based models				
	TXL (trans)	**14.4**	**77.3**	81.1
Ours	DTG-eager	15.5	75.2	-
	DTG-swift	15.0	76.2	-
	DTG	14.9	76.1	**83.9**