Extracurricular Materials

Transformer Grammars

Syntactic structures

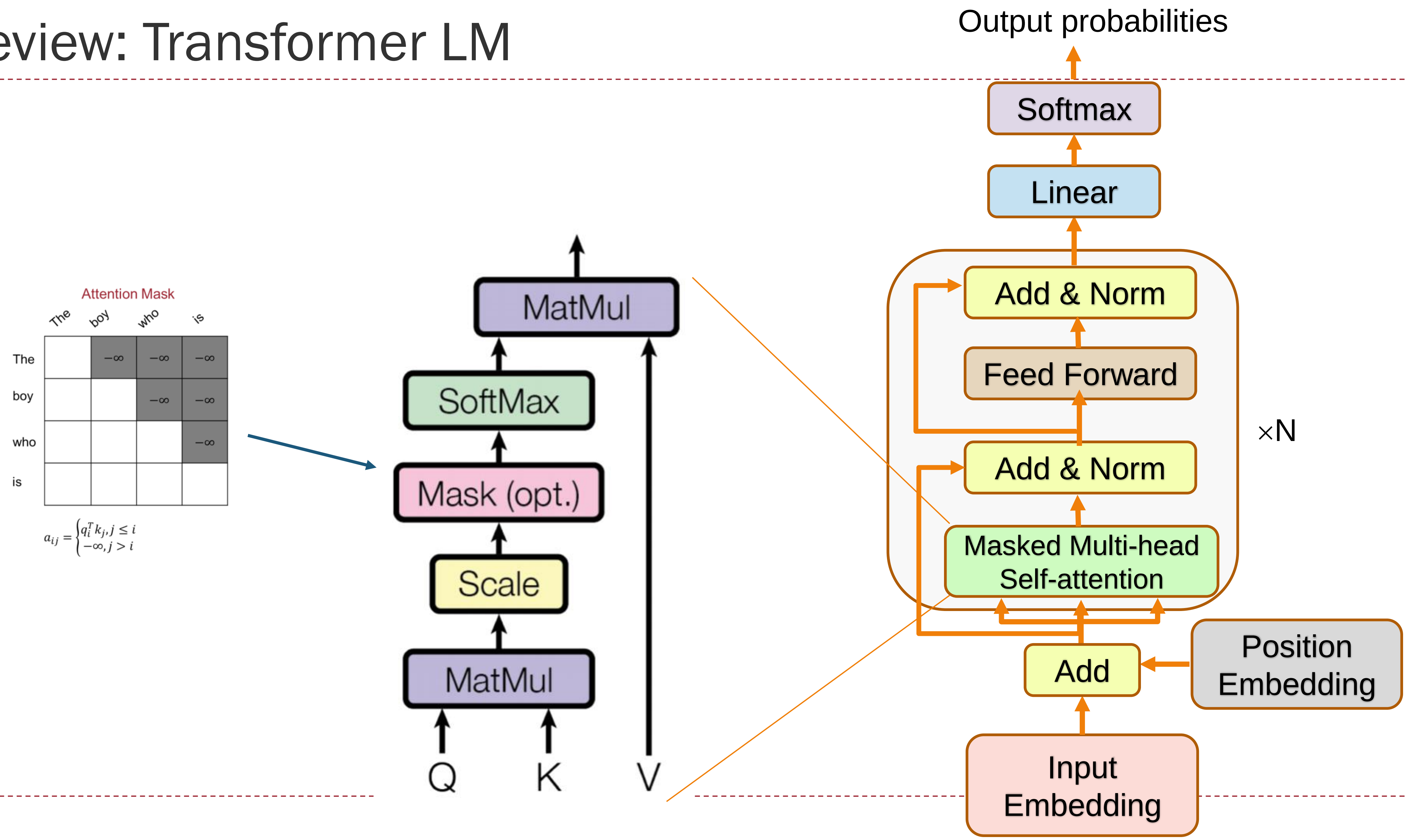
- Past
  - Syntactic structures were deems essential in NLP
- Present
  - NLP dominated by neural models that do not explicitly model syntactic structures
    - In particular, Transformer LMs

- Intuitively, syntactic structures are an intrinsic property of languages and should be helpful
- Can we improve Transformer LMs' performance by modeling syntactic structures?

Transformer Grammars (TG)

- TG is a syntactic language model
  - Jointly model the probability of syntax tree y and words x, i.e., $p(x, y)$
- Idea
  - Generative transition-based parsing
  - Use a Transformer to model the sequence of transitions
    - Actions are implemented through attention masks
  - Goal: encourage the model to explain text through syntax

Review: Transformer LM



Review: Transition-Based Parsing

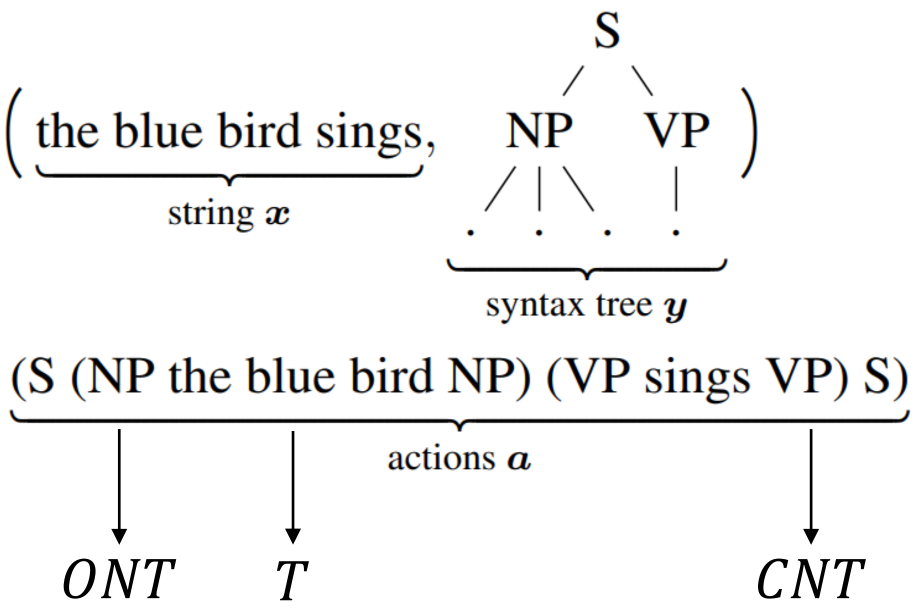
- A parse tree represented as a linear sequence of transitions.
- Parser configuration
  - Buffer B: unprocessed words of the input sentence
  - Stack S: parse tree under construction
- Transition: executing a simple action to transfer one parser configuration to another.

Review: Transition-Based Parsing

- Initial Configuration
  - Buffer B contains the complete input sentence and stack S is empty.
- During parsing
  - Apply a classifier to decide which transition to take next.
  - No backtracking.
- Final Configuration
  - Buffer B is empty and stack S contains the entire parse tree.

Generative transition-based parsing

- A sequence of transitions that simultaneously generate (x, y)
  - No buffer!
- Three types of transitions/actions
  - Opening-nonterminal: ONT
    - Predict a nonterminal node
    - Ex: (NP, (VP
  - Terminal symbol / leaf node: T
    - Predict a terminal word
    - Ex: blue, bird
  - Closing-nonterminal: CNT
    - Close a nonterminal node
    - Ex: NP), VP)



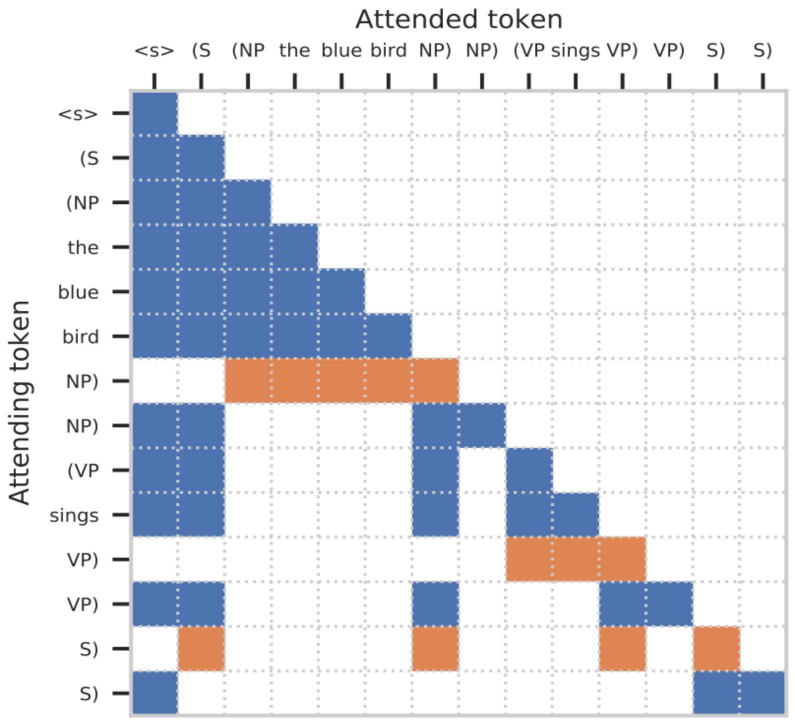
Transformer Grammar

- Both action predictions and stack operations are performed by a Transformer
  - Hence the name “Transformer Grammar” (TG)
- Two types of operations in TG
  - $\boldsymbol{STACK}$
    - Attend to everything on the stack & predict the next action
  - $\boldsymbol{COMPOSE}$
    - Only when the current action is CNT
    - Pop nodes from stack until popping the corresponding ONT
    - Attend to everything popped and compute a node composed from them
    - Push the composed node back to stack
    - No next action prediction

Transformer Grammar

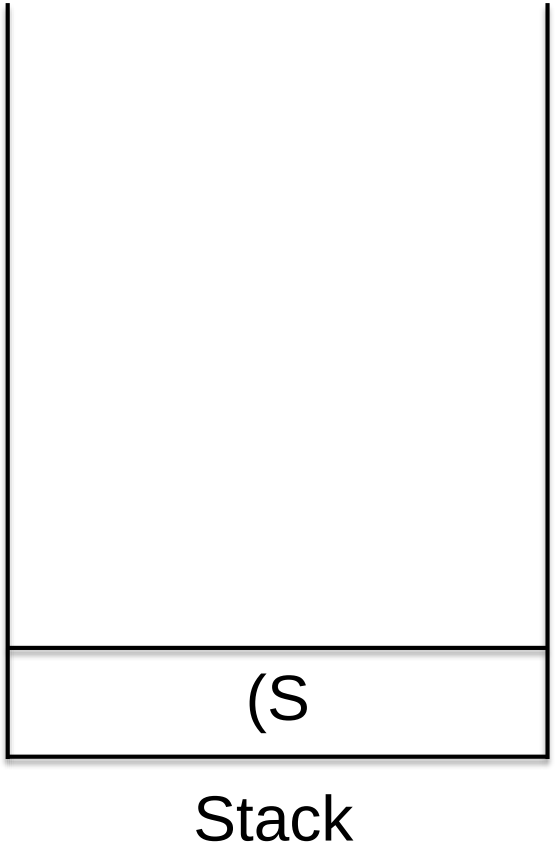
- ONT & T perform $\boldsymbol{STACK}$
- CNT performs both $\boldsymbol{COMPOSE}$ and $\boldsymbol{STACK}$
  - Duplicate CNT as $CNT1$ for $\boldsymbol{COMPOSE}$ and $CNT2$ for $\boldsymbol{STACK}$

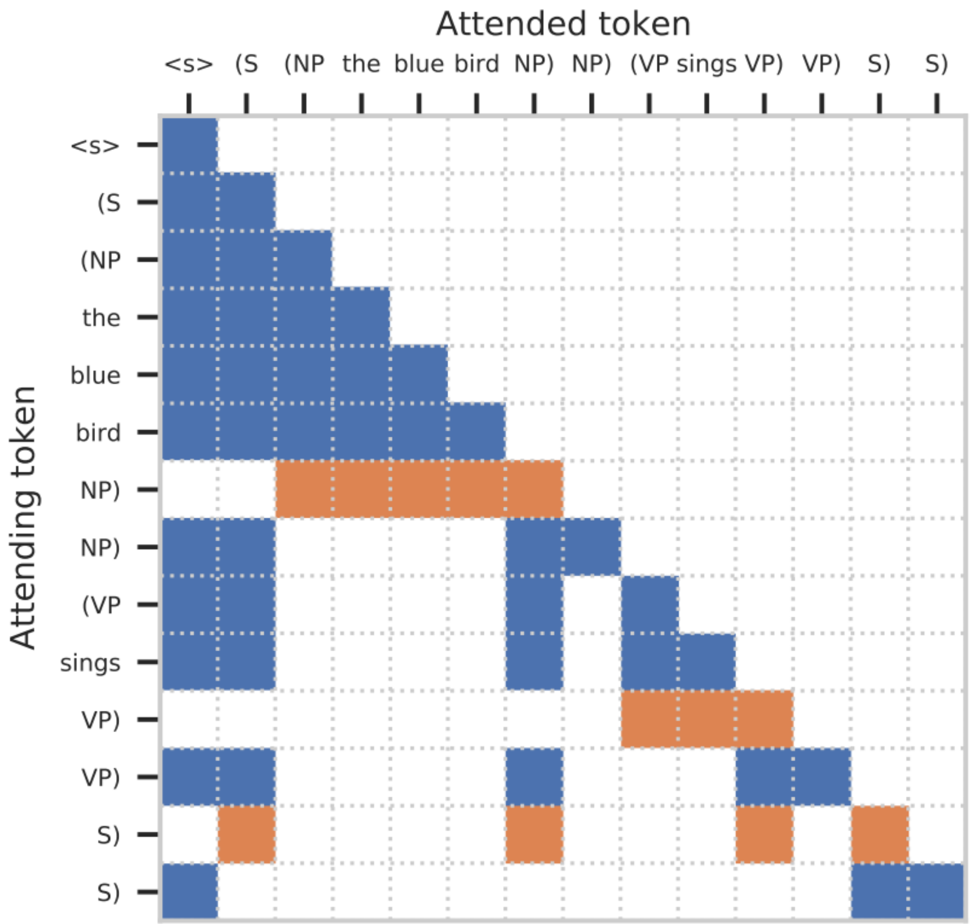
i	Input a'_i	Type	Attn. op.	Label
0	<s>	ONT	STACK	(S
1	(S	ONT	STACK	(NP
2	(NP	ONT	STACK	the
3	the	T	STACK	blue
4	blue	T	STACK	bird
5	bird	T	STACK	NP)
6	NP)	CNT1	COMPOSE	–
7	NP)	CNT2	STACK	(VP
8	(VP	ONT	STACK	sings
9	sings	T	STACK	VP)
10	VP)	CNT1	COMPOSE	–
11	VP)	CNT2	STACK	S)
12	S)	CNT1	COMPOSE	–
13	S)	CNT2	STACK	–



Example

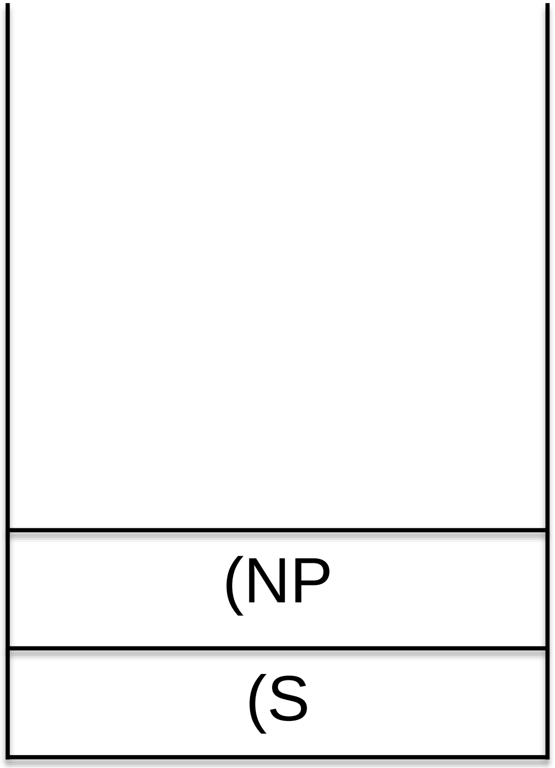
- Action sequence: (S



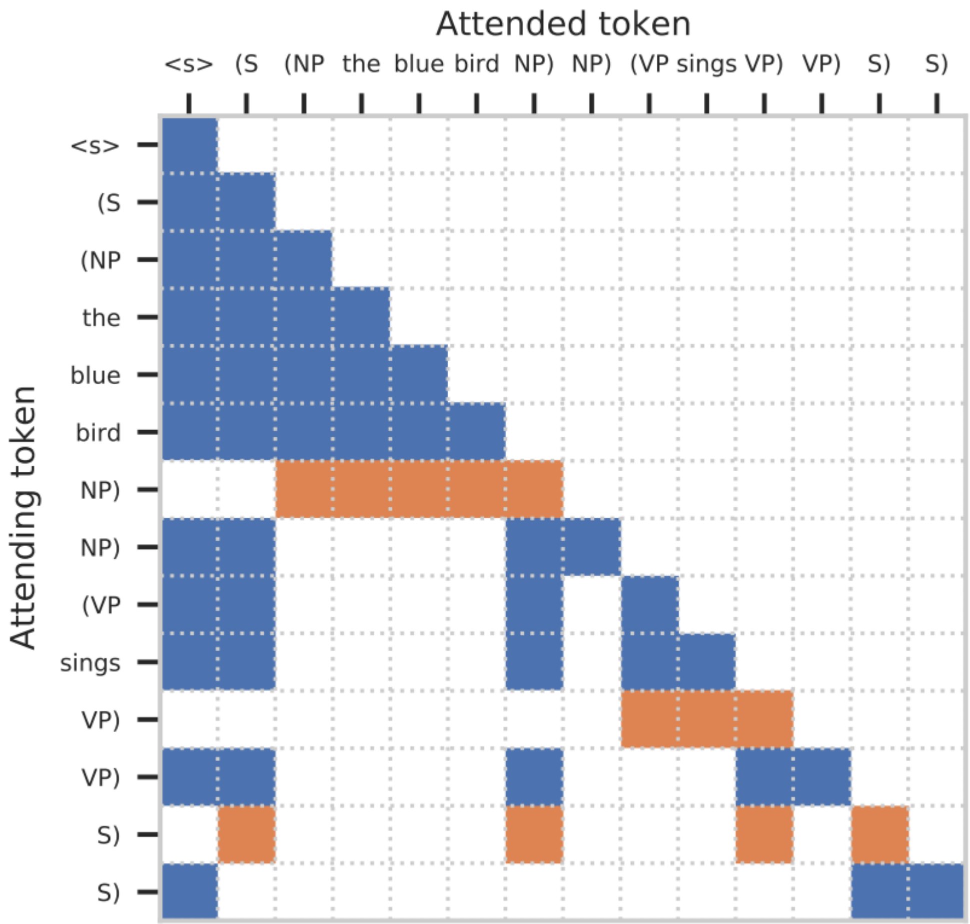


Example

- Action sequence: (S (NP

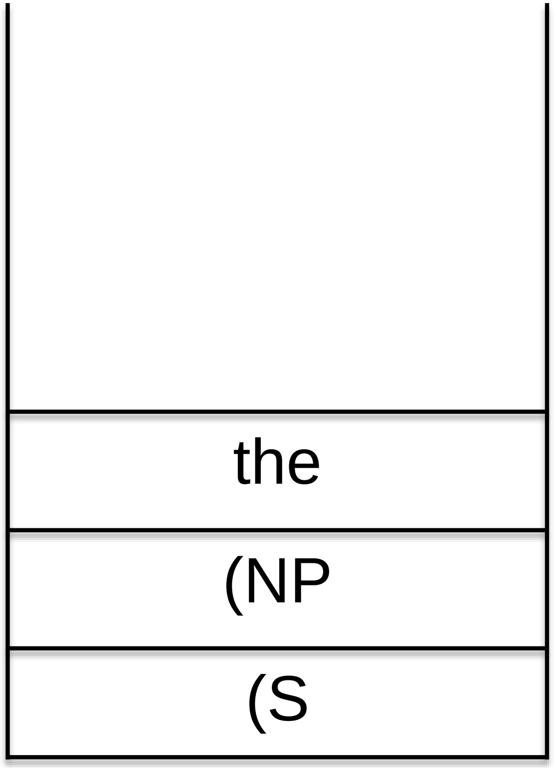


Stack

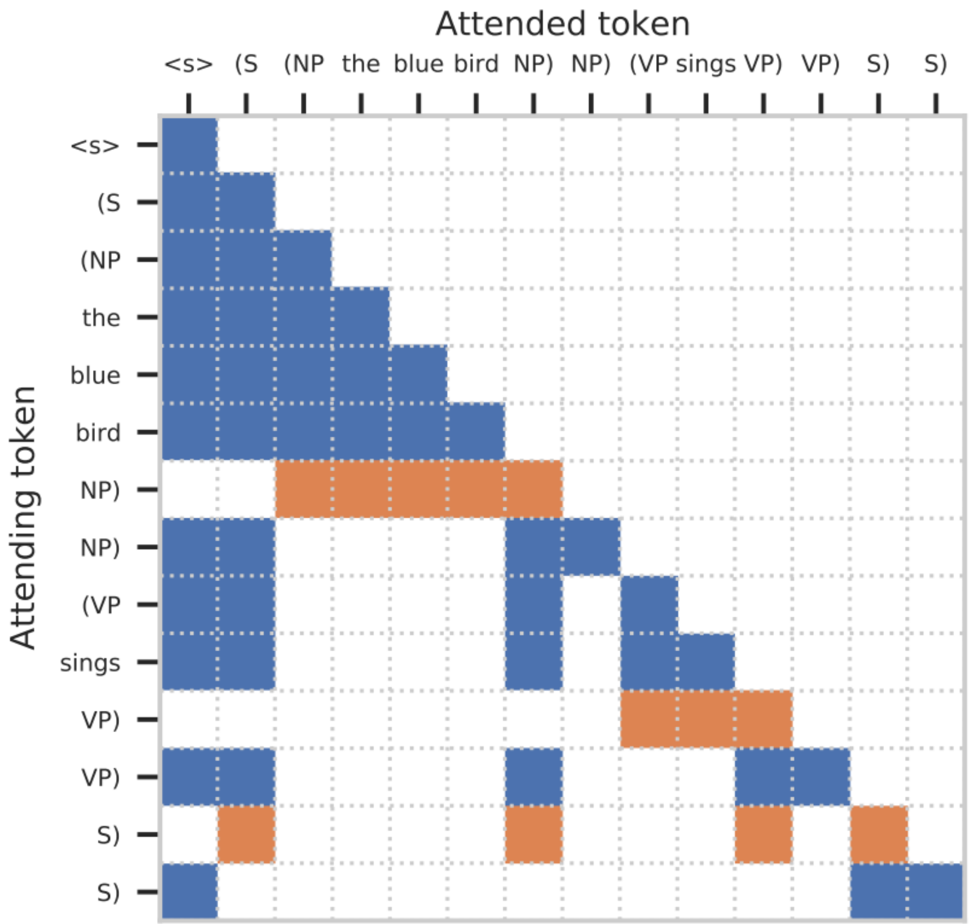


Example

- Action sequence: (S (NP the

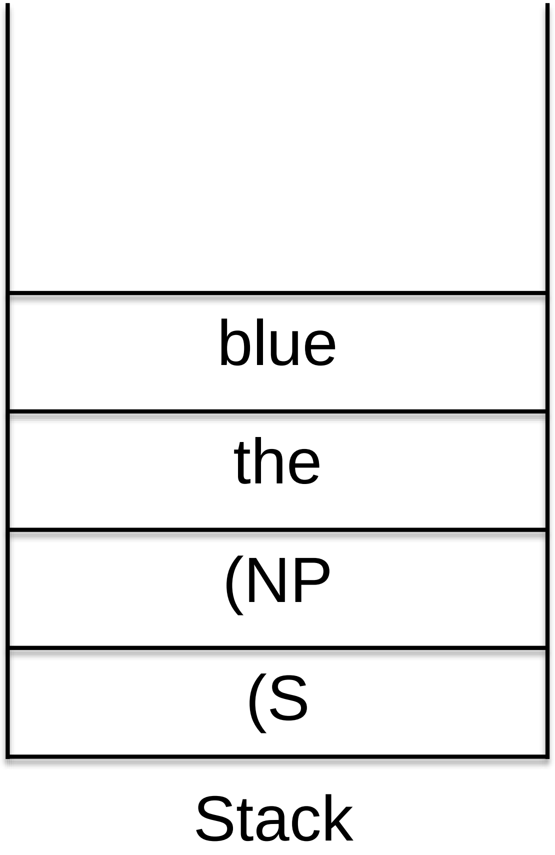


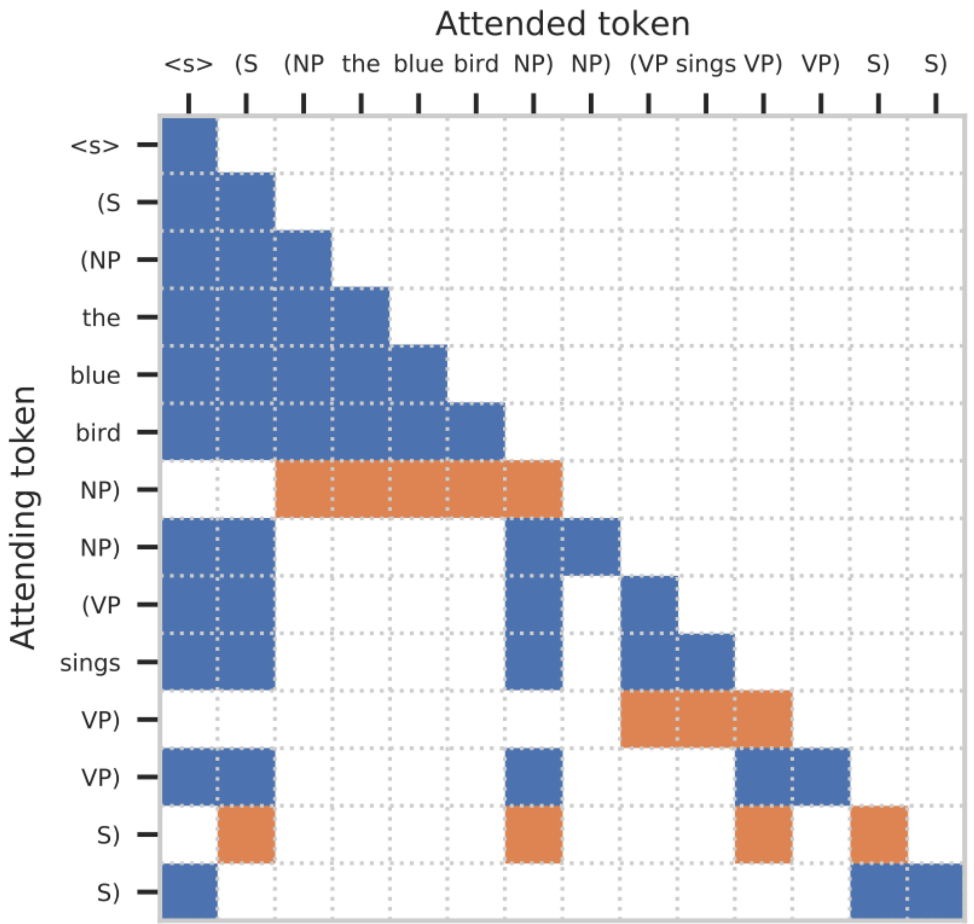
Stack



Example

- Action sequence: (S (NP the blue

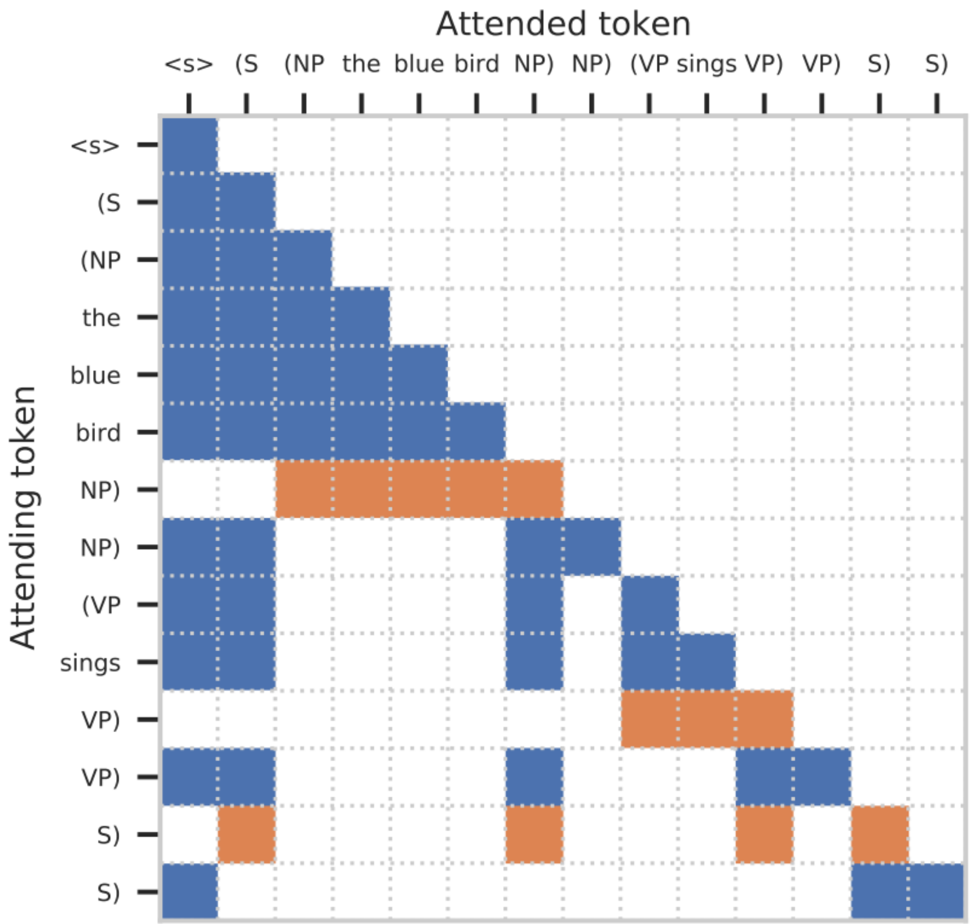




Example

- Action sequence: (S (NP the blue bird

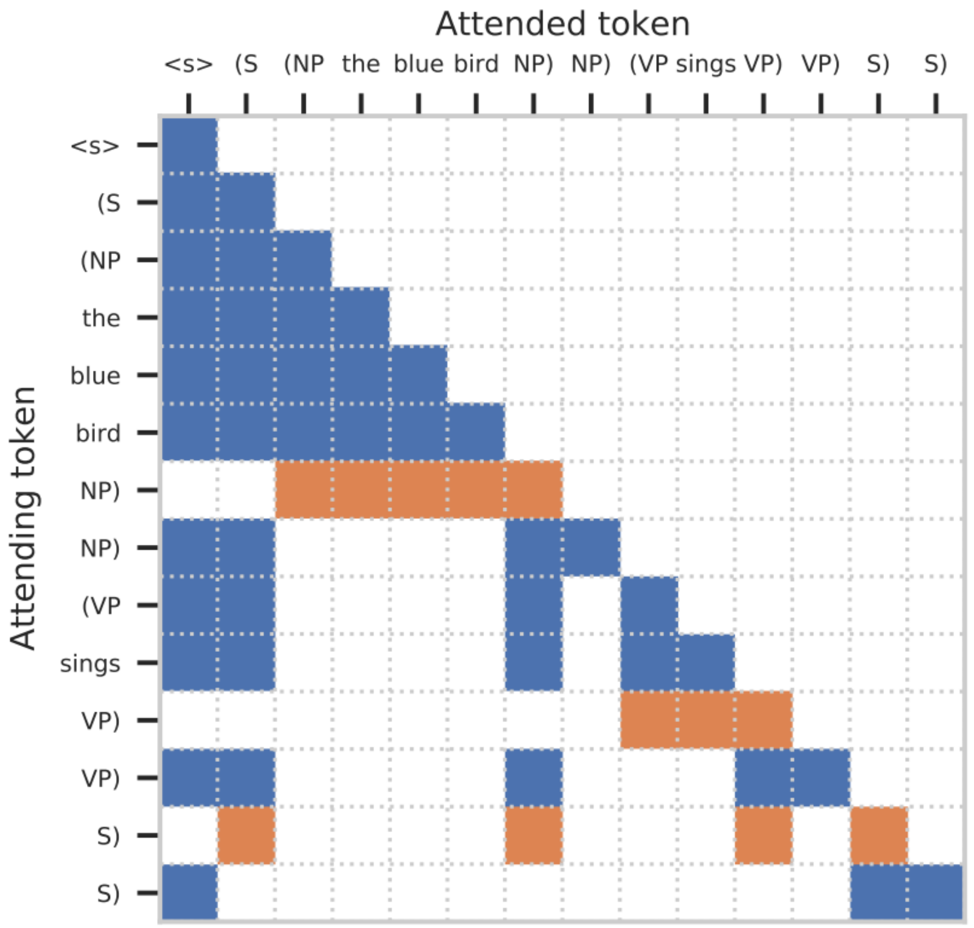
Stack



Example

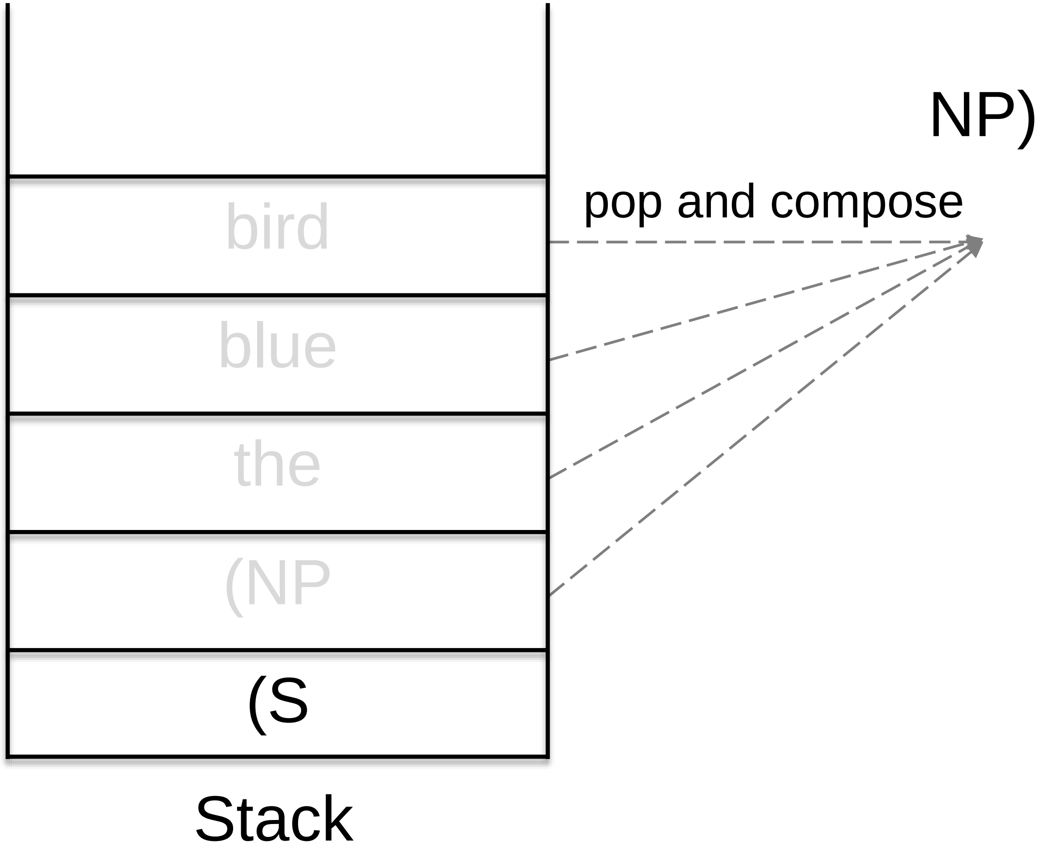
- Action sequence: (S (NP the blue bird NP)

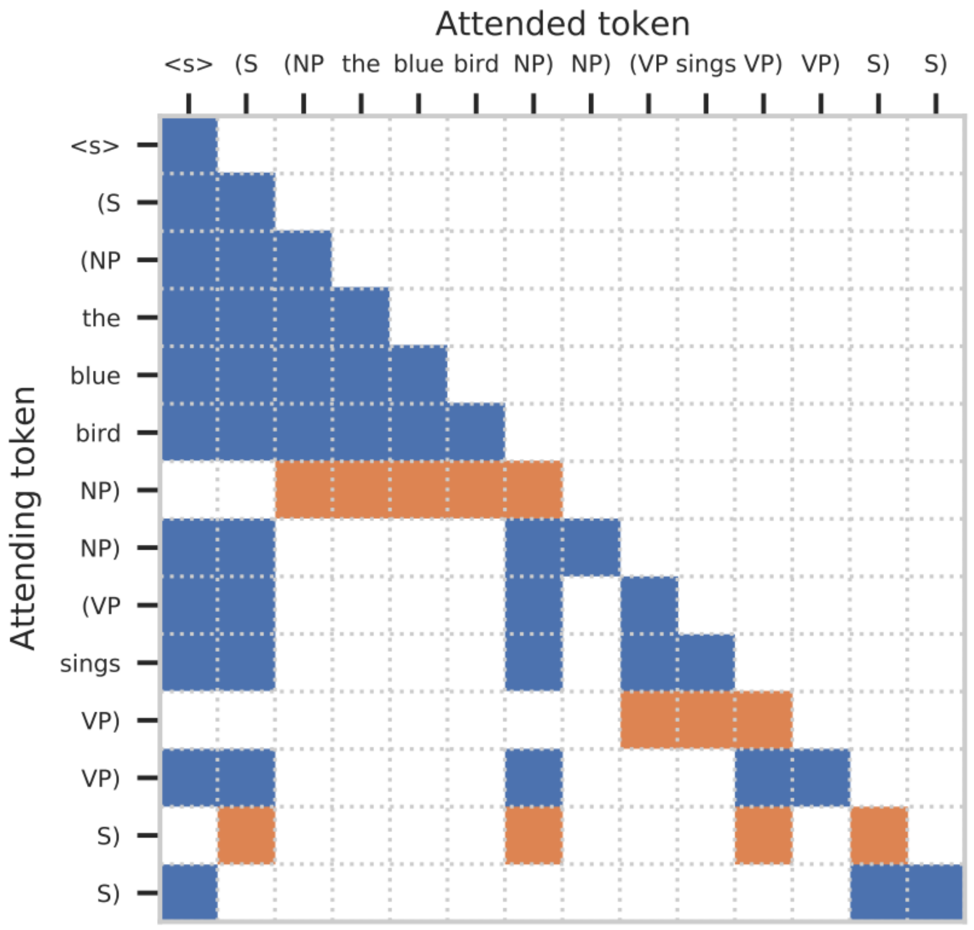
Stack



Example

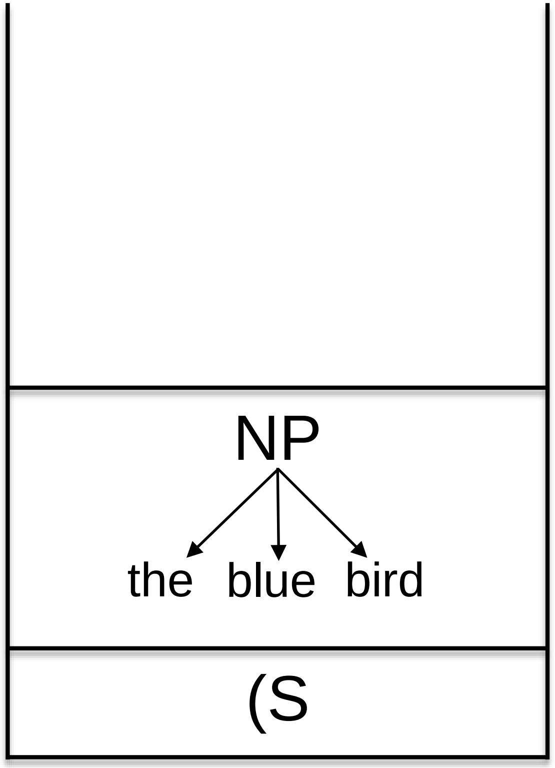
- Action sequence: (S (NP the blue bird NP)



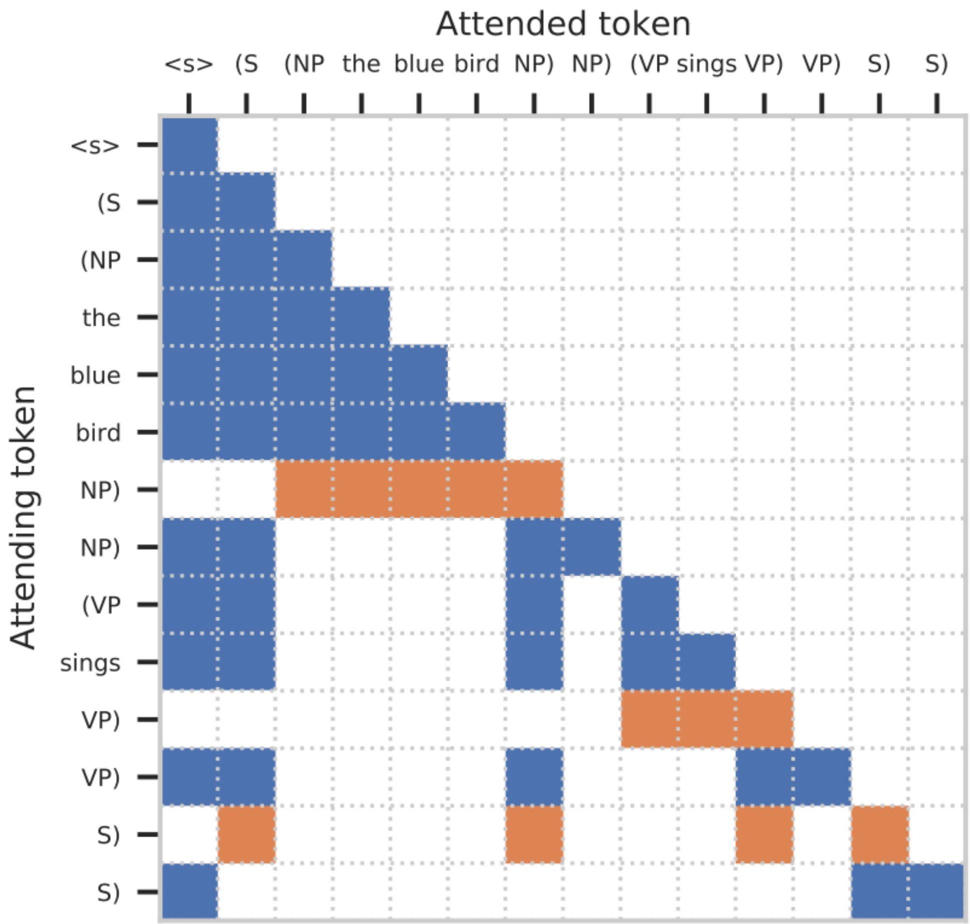


Example

- Action sequence: (S (NP the blue bird NP)

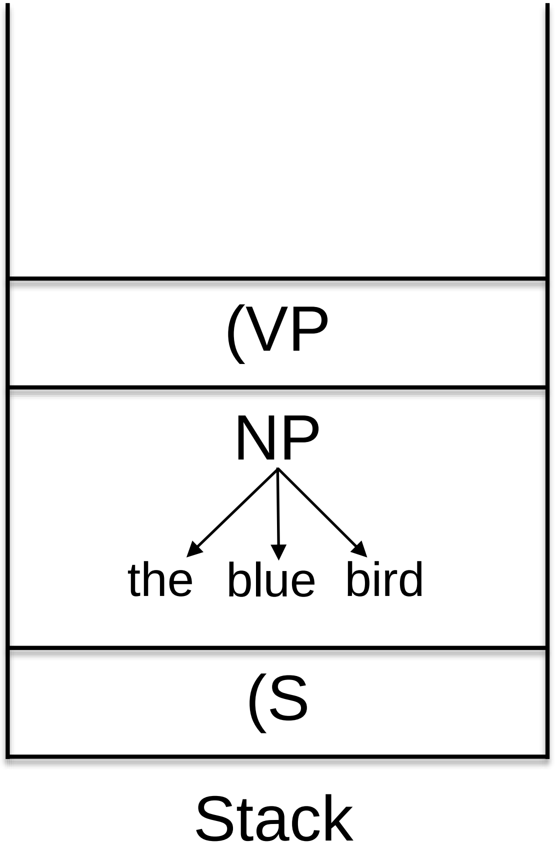


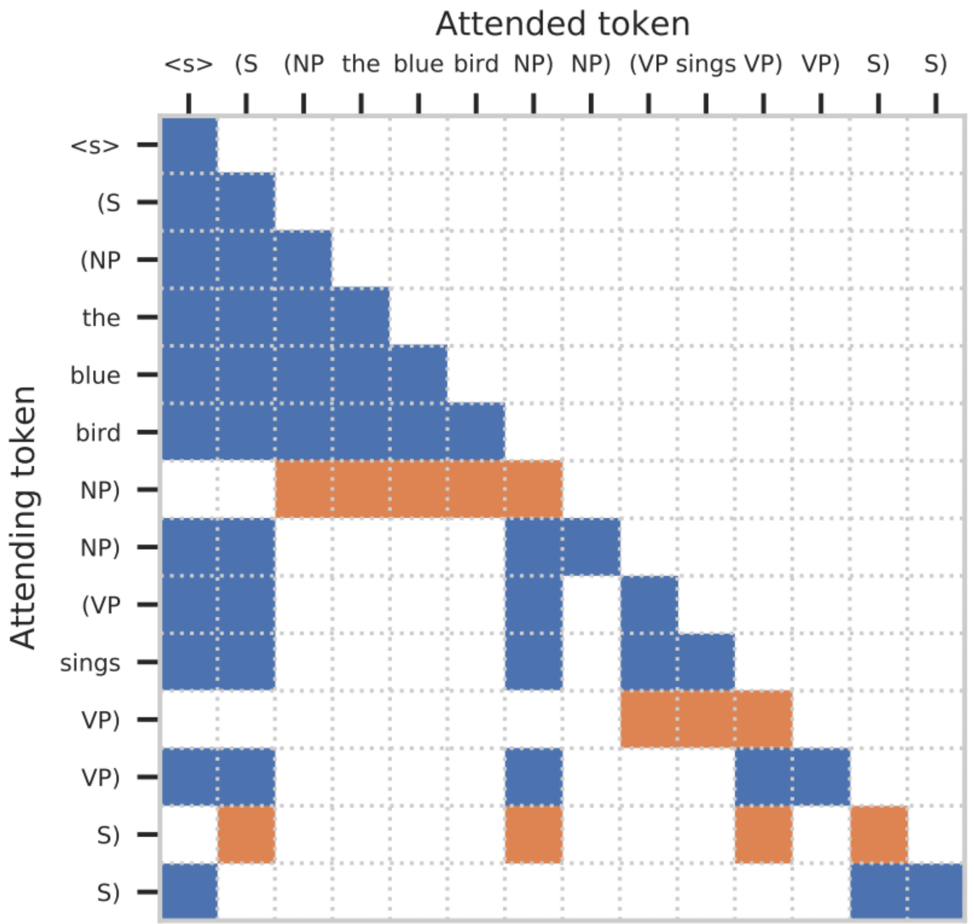
Stack



Example

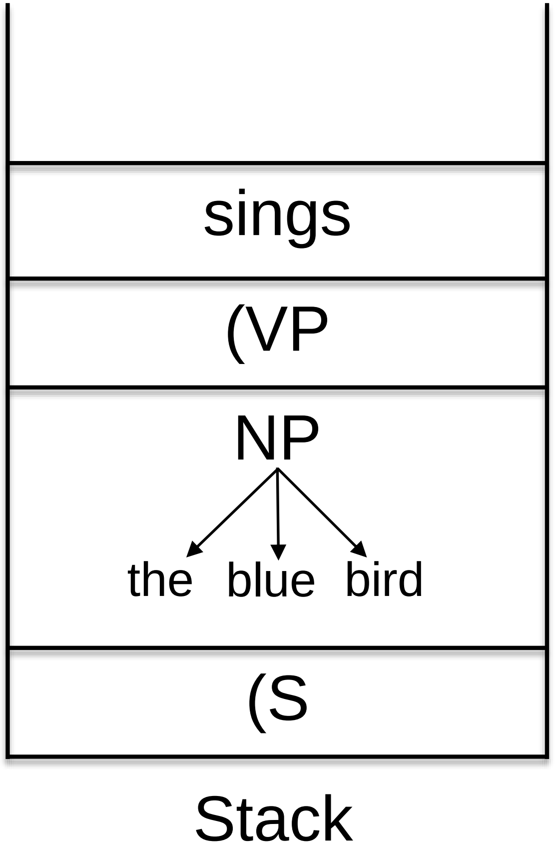
- Action sequence: (S (NP the blue bird NP) (VP

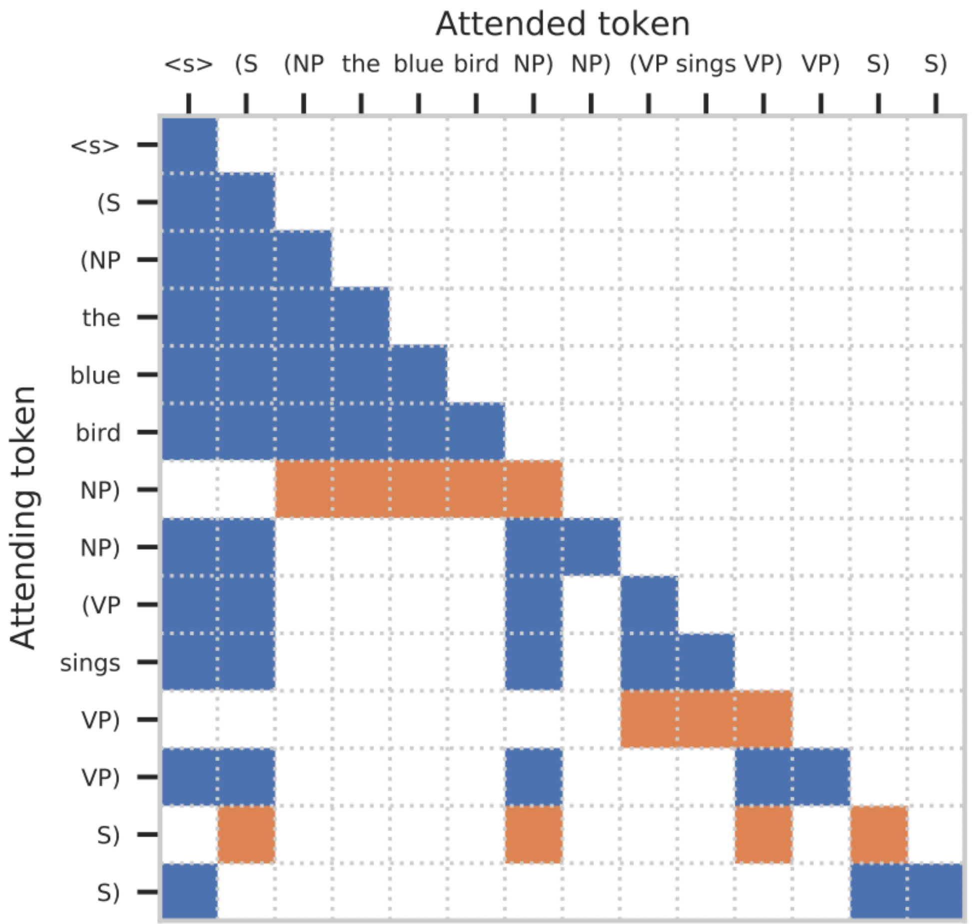




Example

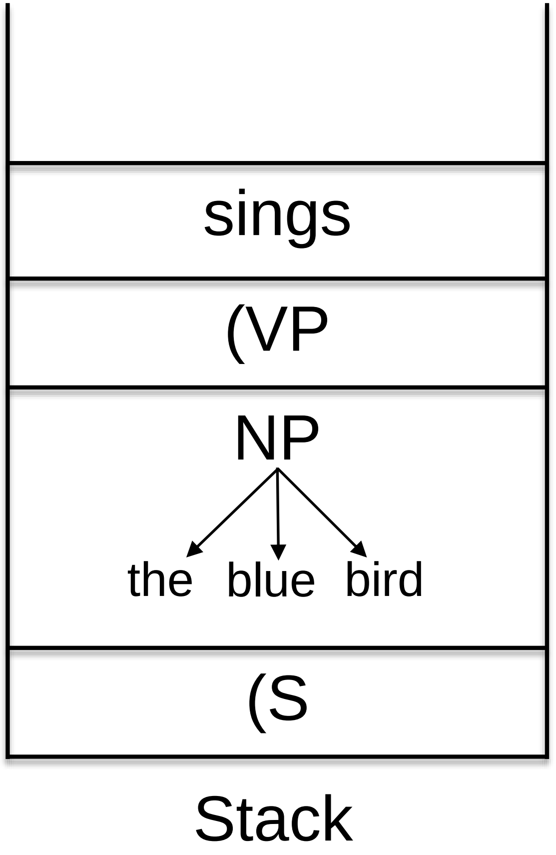
- Action sequence: (S (NP the blue bird NP) (VP sings

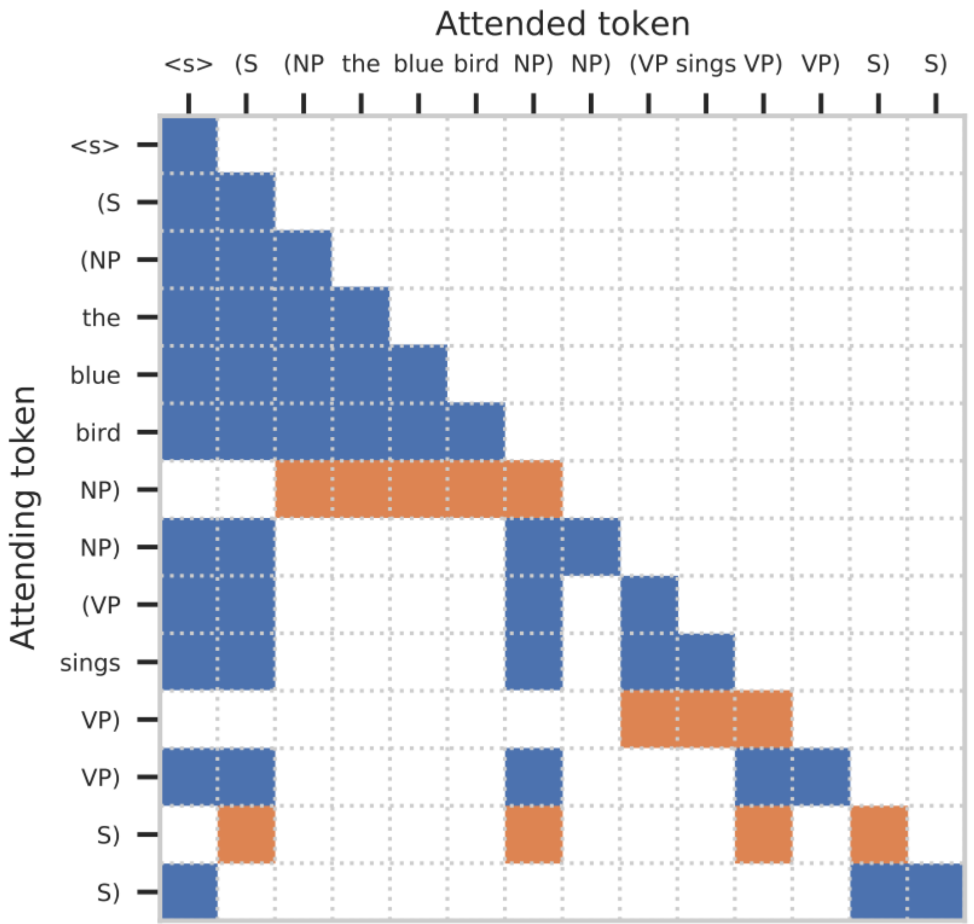




Example

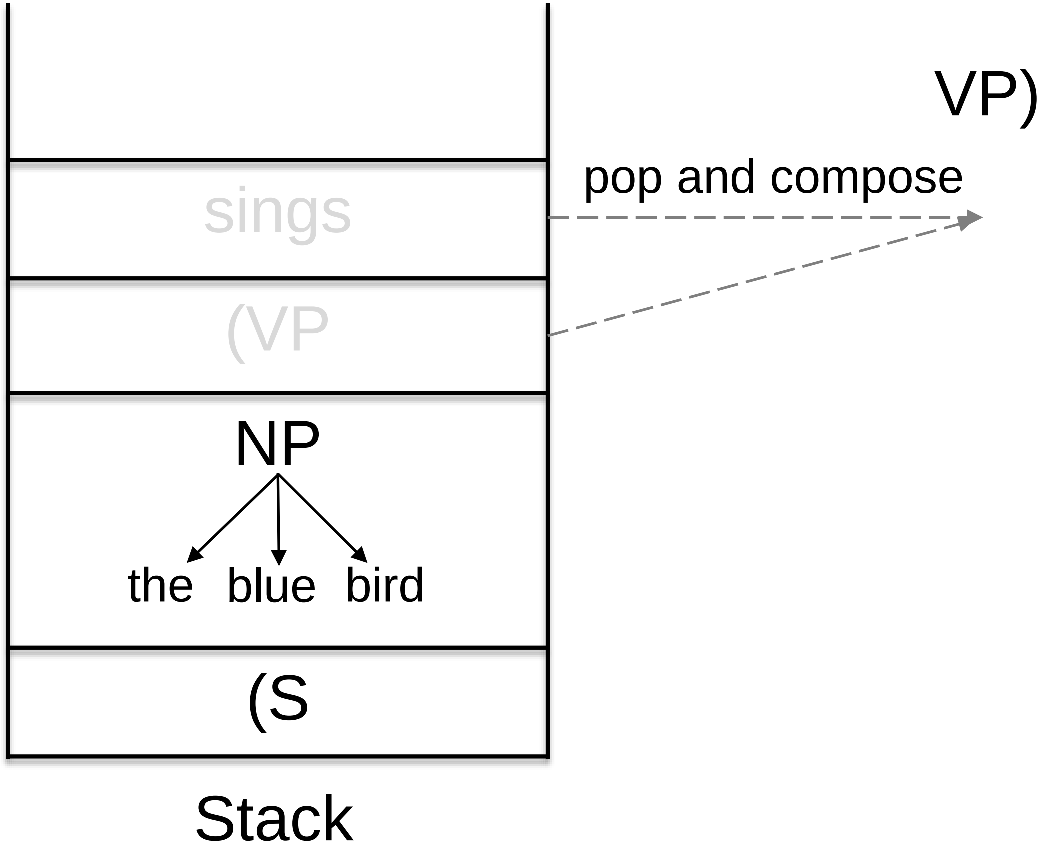
- Action sequence: (S (NP the blue bird NP) (VP sings VP)

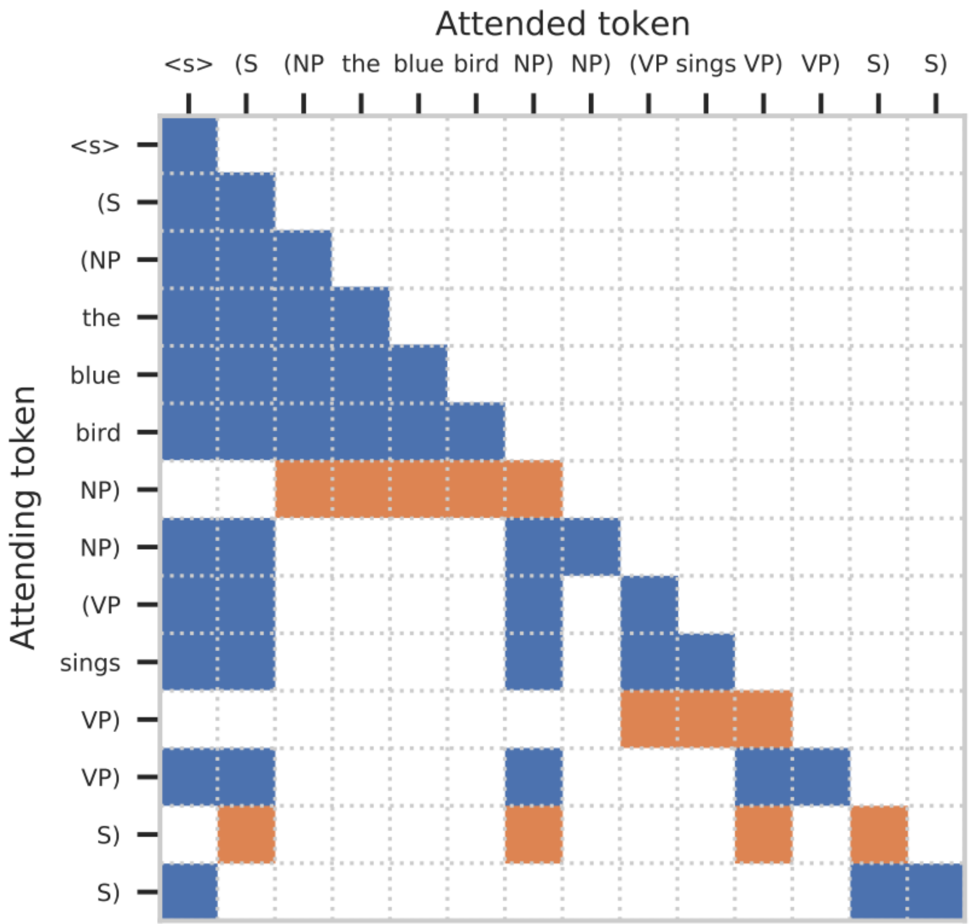




Example

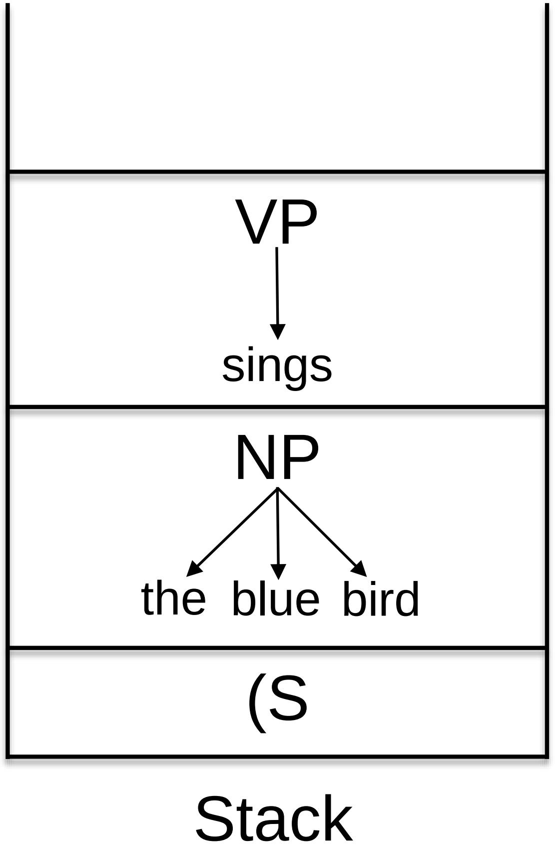
- Action sequence: (S (NP the blue bird NP) (VP sings VP)

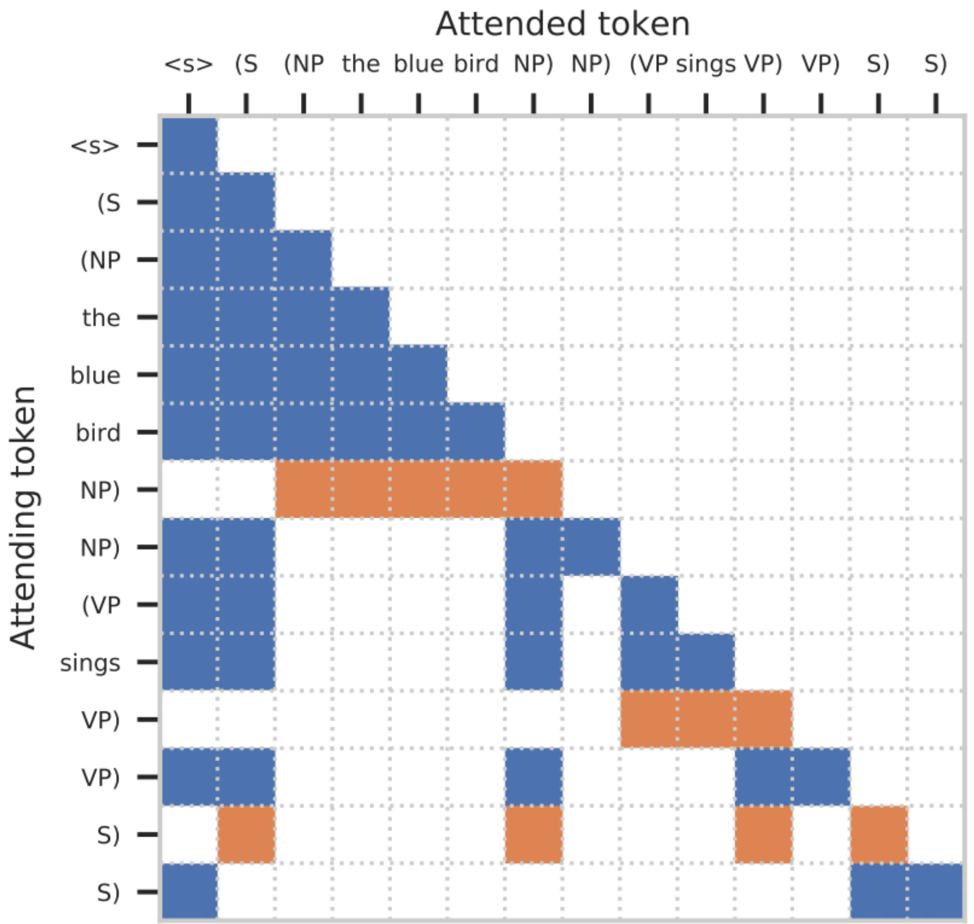




Example

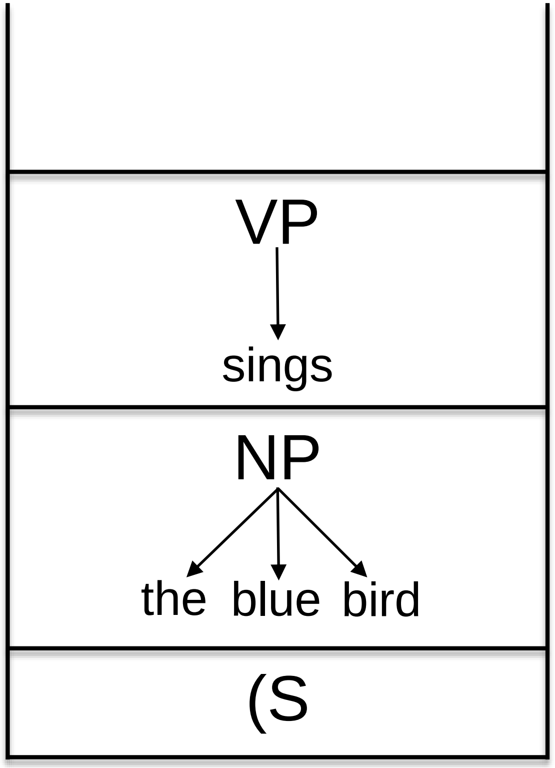
- Action sequence: (S (NP the blue bird NP) (VP sings VP)



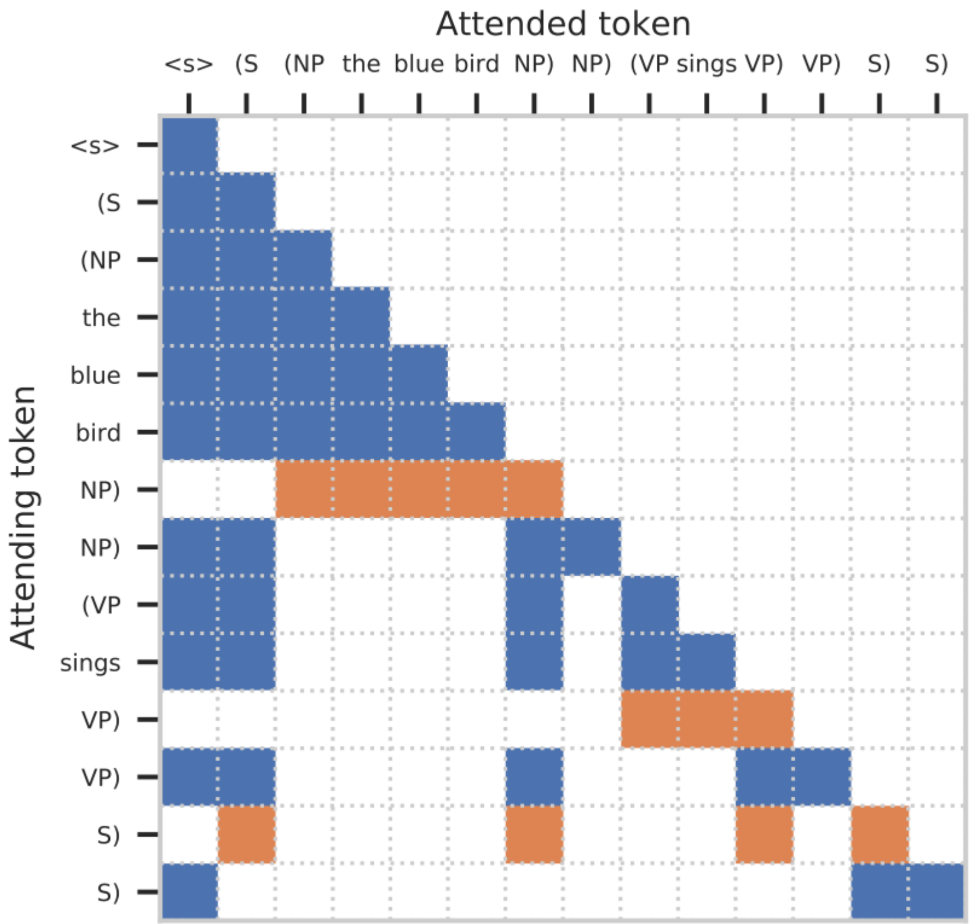


Example

- Action sequence: (S (NP the blue bird NP) (VP sings VP) S)

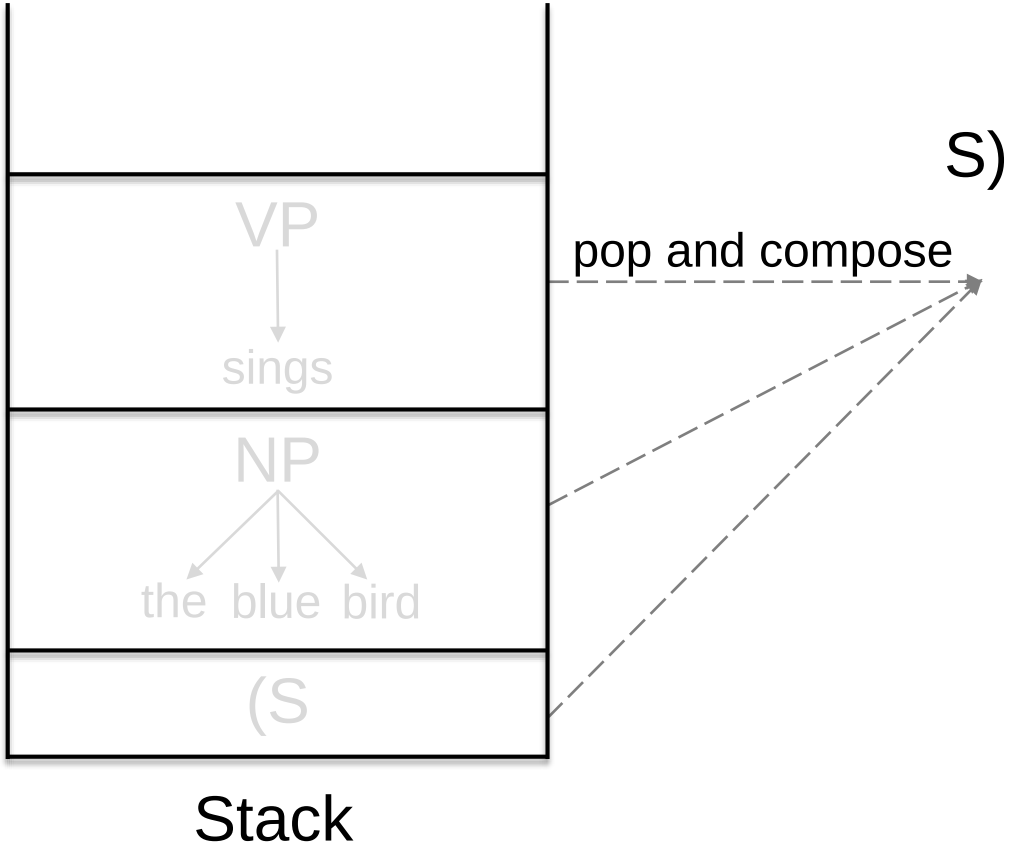


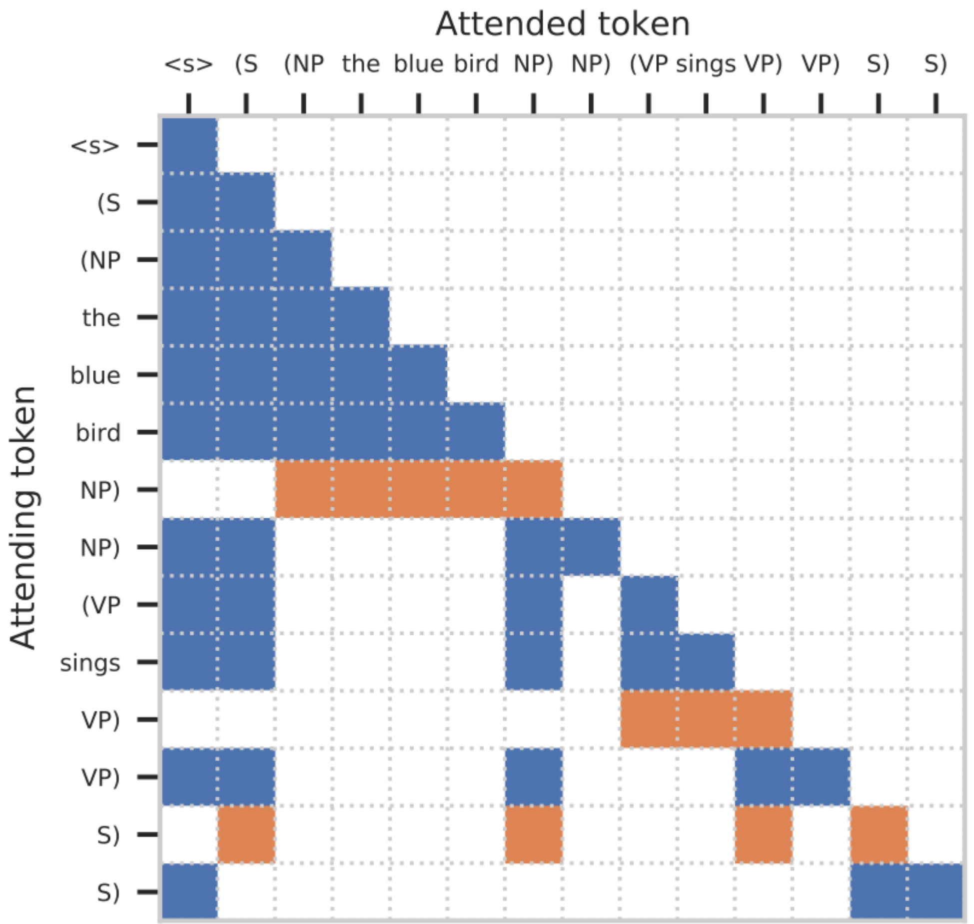
Stack



Example

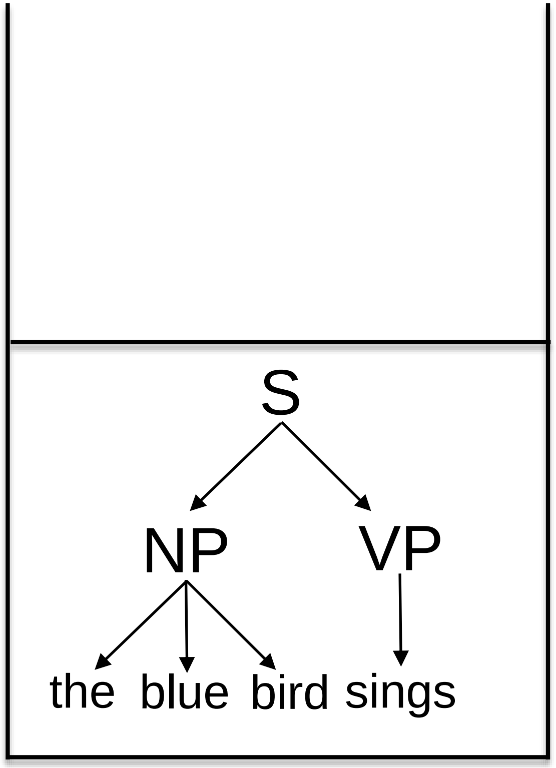
- Action sequence: (S (NP the blue bird NP) (VP sings VP) S)



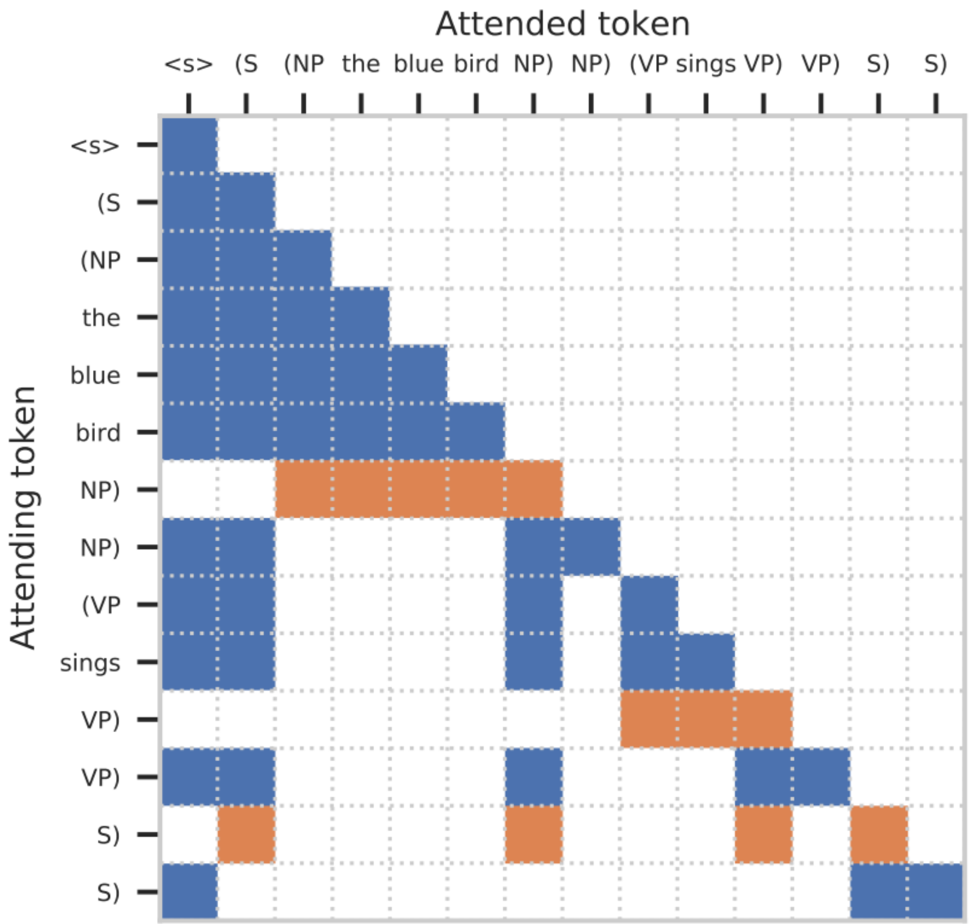


Example

- Action sequence: (S (NP the blue bird NP) (VP sings VP) S)



Stack



Experiments

- Language Modeling
  - TG models the joint distribution $p(x, y)$ of the syntactic tree y and words x.
    - Hard to compute $p(x) = \sum_y p(x, y)$.
  - Compute $p(x) \approx \sum_{y \in Y} p(x, y)$ to get a probability lower bound.
    - Y is a set of best syntax trees of sentence x given by a predefined parser.
  - Baselines
    - TXL (trees): naïve transformer trained on the action sequence
      - No specially design masking
    - TXL (terminals): trained on the words only.

	Perplexity (↓)		
	PTB	BLLIP sent.	BLLIP doc.
TG†	61.8 ± 0.2	30.3 ± 0.5	26.3 ± 0.1
TXL (trees)†	**61.2 ± 0.3**	**29.8 ± 0.4**	**22.1 ± 0.1**
TXL (terminals)	62.6 ± 0.2	31.2 ± 0.4	23.1 ± 0.1

Experiments

- Syntactic generalization
  - Syntactically correct sentence vs. incorrect decoy sentence
  - 31 types of sentence pairs.
    - Subject-verb agreement: the number feature of a verb must agree with its upstream
      - P(officers who love the skater smile) > P(officers who love the skater smiles)
    - Negative polarity licensing, *any* is often used in negative sentences.
      - P(No managers have any luck) > P(The managers have any luck)
    - NP/Z Garden-path ambiguity: word following verb may be noun phrases (NP) or None(N)
      - P(As it crossed the sea remained calm) < P(As it crossed, the sea remained calm)
    - Gross Syntactic Expectation, Center Embedding, …

Experiments

- Syntactic generalization
  - Syntactically correct sentence vs. incorrect decoy sentence
- TG is stronger than TXL(trees) and other much larger LMs
  - Syntactic inductive bias enhances its syntactic ability.

		SG (↑) BLLIP sent.
TG†	252M	**82.5 ± 1.6**
TXL (trees)†		80.2 ± 1.6
TXL (terminals)		69.5 ± 2.1
GPT-2 (Radford et al., 2019)		78.4♠
Gopher (Rae et al., 2021)	280B	79.5
Chinchilla (Hoffmann et al., 2022)	70B	79.7

Summary

- Explicit modeling of syntax in transformer LM can be beneficial
  - LM perplexity
  - Syntactic generalization

- Next
  - Dependency syntax?
  - No gold tree?
  - Downstream tasks?

Papers

- “Transformer Grammars: Augmenting Transformer Language Models with Syntactic Inductive Biases at Scale” TACL 2022
- Related work:
  - “Pushdown Layers: Encoding Recursive Structure in Transformer Language Models” EMNLP 2023
  - “Dependency Transformer Grammars: Integrating Dependency Structures into Transformer Language Models” Under review
  - “Generative Pretrained Structured Transformers: Unsupervised Syntactic Language Models at Scale” Under review