

Should Neural Network Architecture Reflect Linguistic Structure?

Chris Dyer (CMU → Google/DeepMind)

Joint work with:

Miguel Ballesteros (UPF)

Wang Ling (Google/DeepMind)

Austin Matthews (CMU)

Noah A. Smith (UW)

Learning language

ARBITRARINESS (de Saussure, 1916)

COMPOSITIONALITY (Frege, 1892)

Learning language

ARBITRARINESS (de Saussure, 1916)

car – c + b = bar



COMPOSITIONALITY (Frege, 1892)

Learning language

ARBITRARINESS (de Saussure, 1916)

car - c + b = **bar**



cat - c + b = **bat**



COMPOSITIONALITY (Frege, 1892)

Learning language

ARBITRARINESS (de Saussure, 1916)

car - c + b = **bar**



cat - c + b = **bat**



car



COMPOSITIONALITY (Frege, 1892)

Learning language

ARBITRARINESS (de Saussure, 1916)

car – c + b = bar



cat – c + b = bat



car

Auto

voiture

xe hơi



ọkọ ayọkẹlẹ

koloi

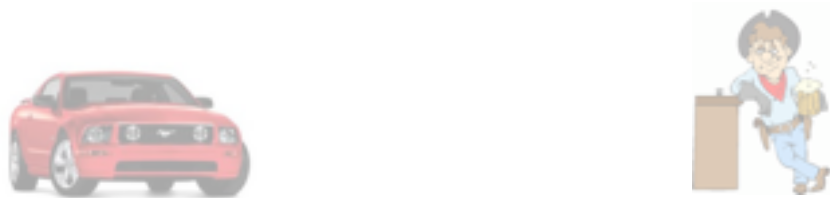
sakyanan

COMPOSITIONALITY (Frege, 1892)

Learning language

ARBITRARINESS (de Saussure, 1916)

car – c + b = bar



cat – c + b = bat



car

Auto

voiture

xe hơi



ọkọ ayọkẹlẹ

koloi

sakyanan

COMPOSITIONALITY (Frege, 1892)

Learning language

ARBITRARINESS (de Saussure, 1916)

car – c + b = bar



cat – c + b = bat



car

Auto

voiture

xe hơi



ọkọ ayọkẹlẹ

koloi

sakyanan

COMPOSITIONALITY (Frege, 1892)

John dances – John + Mary = Mary dances

DANCE(JOHN)

DANCE(MARY)

Learning language

ARBITRARINESS (de Saussure, 1916)

car – c + b = bar



cat – c + b = bat



car

Auto

voiture

xe hơi



ọkọ ayọkẹlẹ

koloi

sakyanan

COMPOSITIONALITY (Frege, 1892)

John dances – John + Mary = Mary dances

DANCE(JOHN)

DANCE(MARY)

John sings – John + Mary = Mary sings

SING(JOHN)

SING(MARY)

Learning language

ARBITRARINESS (de Saussure, 1916)

car – c + b = bar



cat – c + b = bat



car

Auto

voiture

xe hơi



ọkọ ayọkẹlẹ

koloi

sakyanan

COMPOSITIONALITY (Frege, 1892)

John dances – John + Mary = Mary dances

DANCE(JOHN)

DANCE(MARY)

John sings – John + Mary = Mary sings

SING(JOHN)

SING(MARY)

Learning language

ARBITRARINESS (de Saussure, 1916)

car – c + b = bar

cat – c + b = bat



Memorize



car

A

xe hơi



ọkọ ayọkẹlẹ

koloi

sakyanan

COMPOSITIONALITY (Frege, 1892)

John dances – John + Mary = Mary dances

DANCE(John)

DANCE(MARY)

John sings

Generalize

sings

SING(John)

SING(MARY)

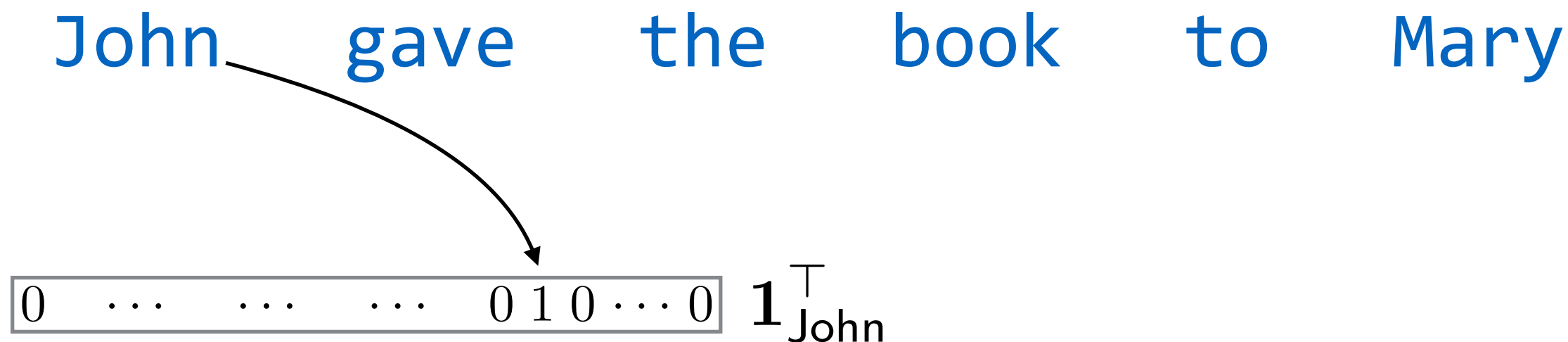
Learning language

John gave the book to Mary

Learning language

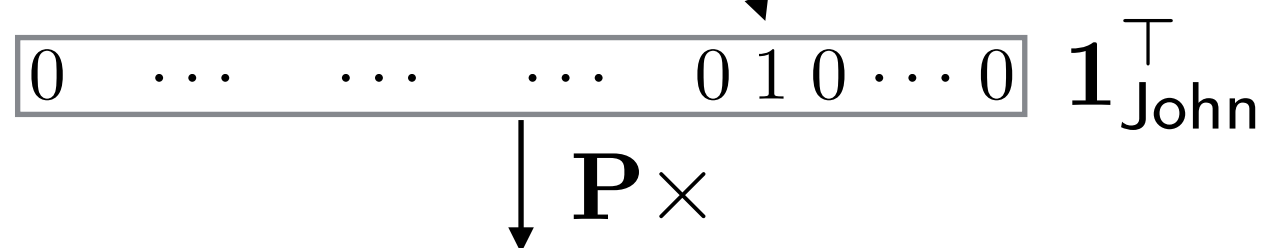
John gave the book to Mary

$[0 \quad \dots \quad \dots \quad \dots \quad 0 \quad 1 \quad 0 \quad \dots \quad 0] \mathbf{1}_{\text{John}}^T$



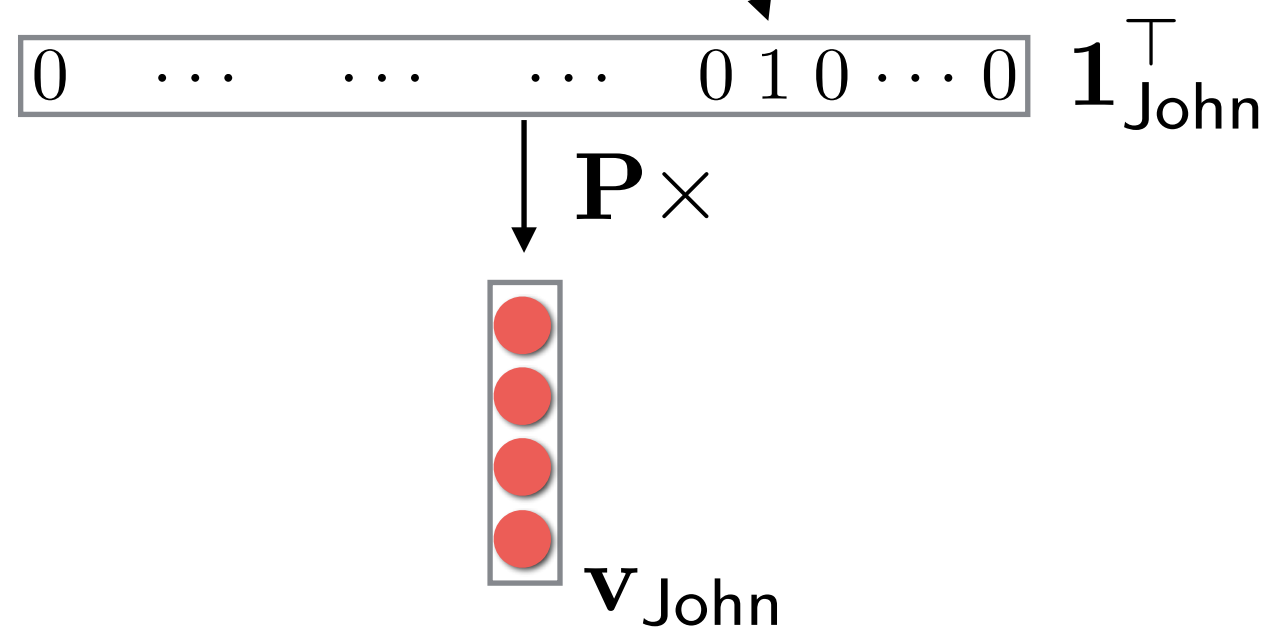
Learning language

John gave the book to Mary



Learning language

John gave the book to Mary



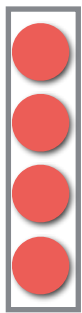
Learning language

John gave the book to Mary



Learning language

John



gave



the



book



to

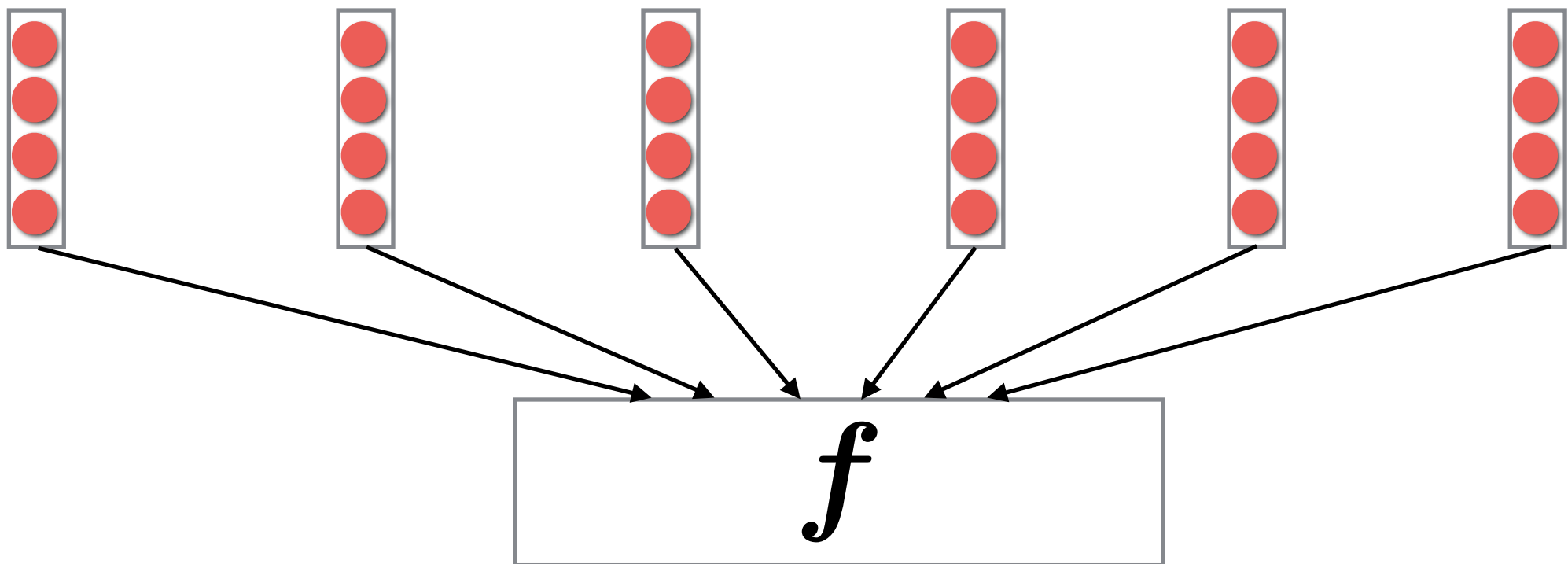


Mary

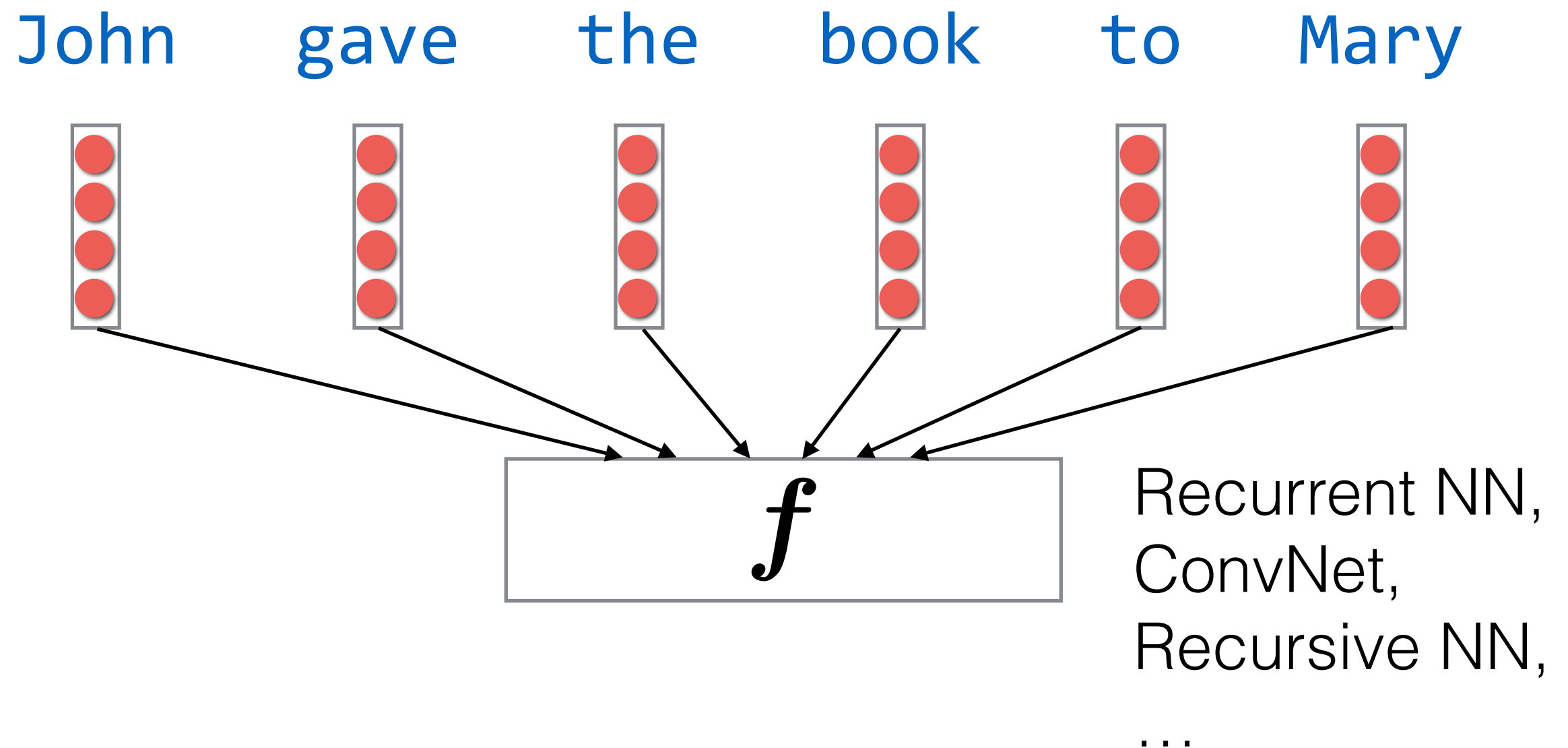


Learning language

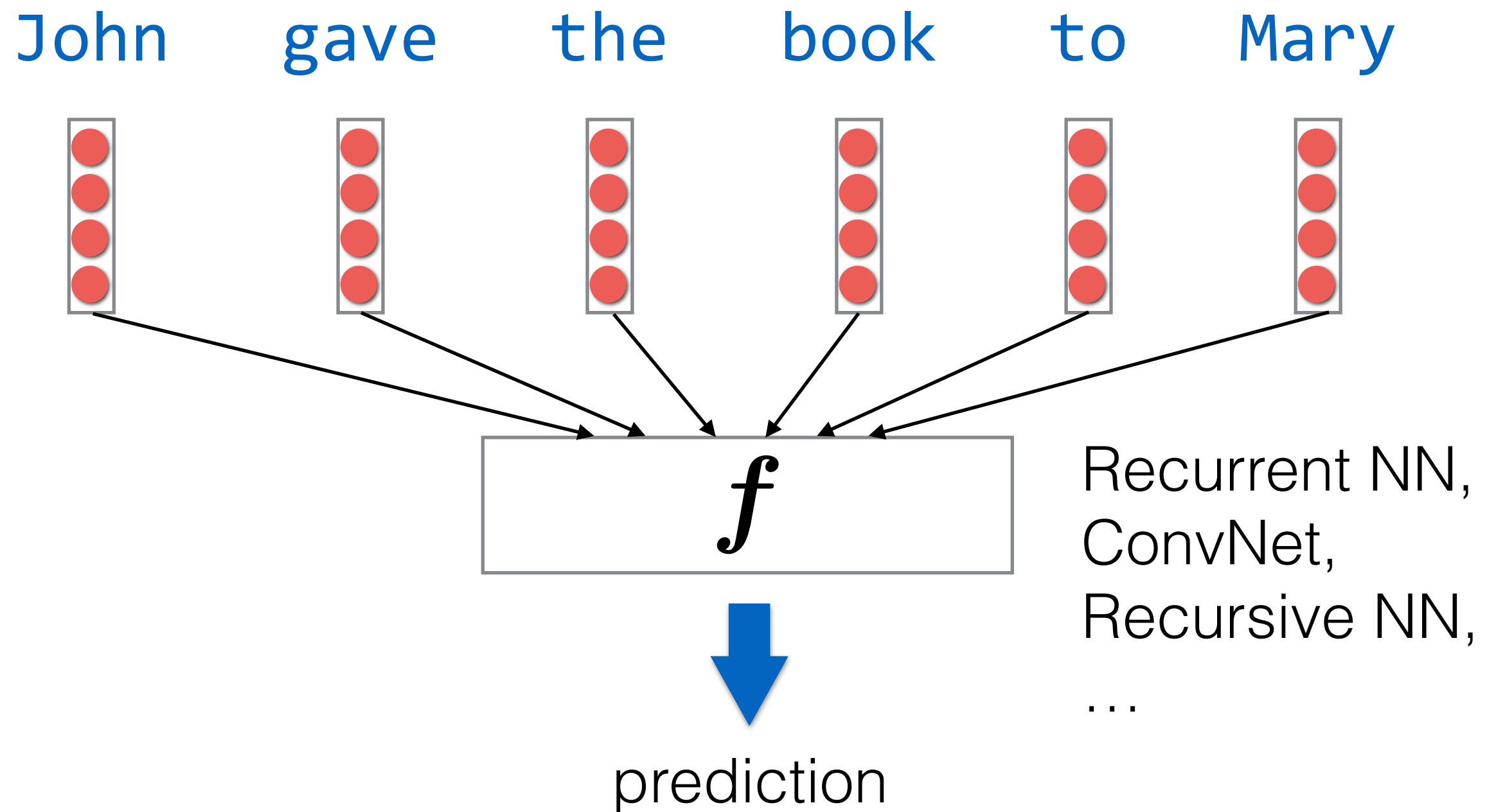
John gave the book to Mary



Learning language



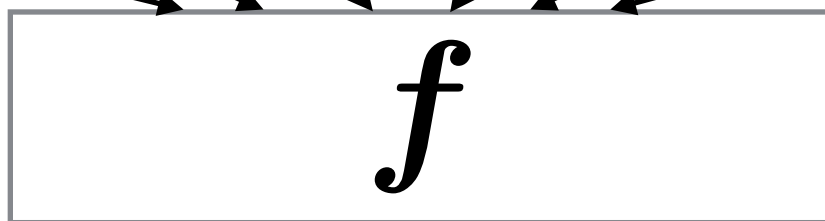
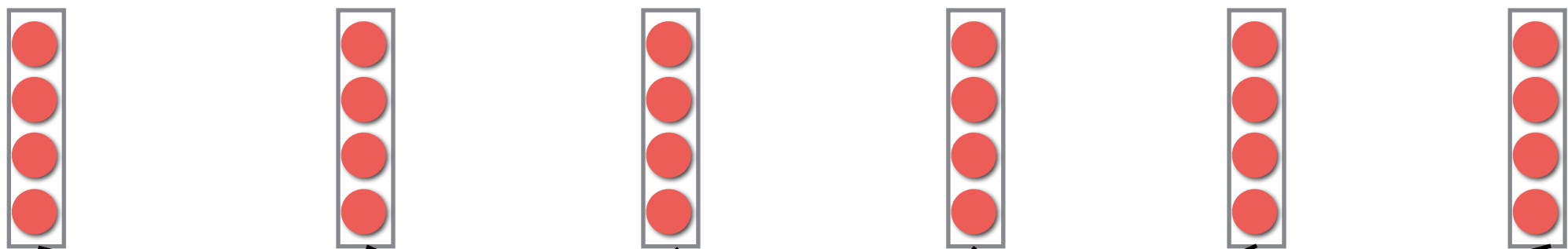
Learning language



Learning language

Memorize

John gave the book to Mary



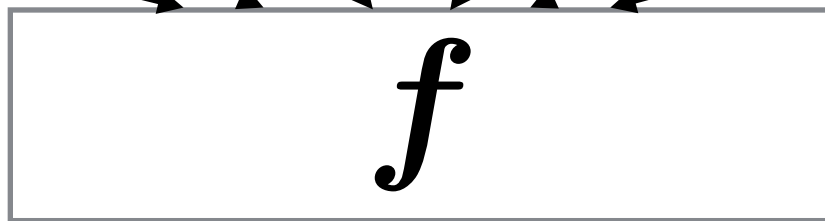
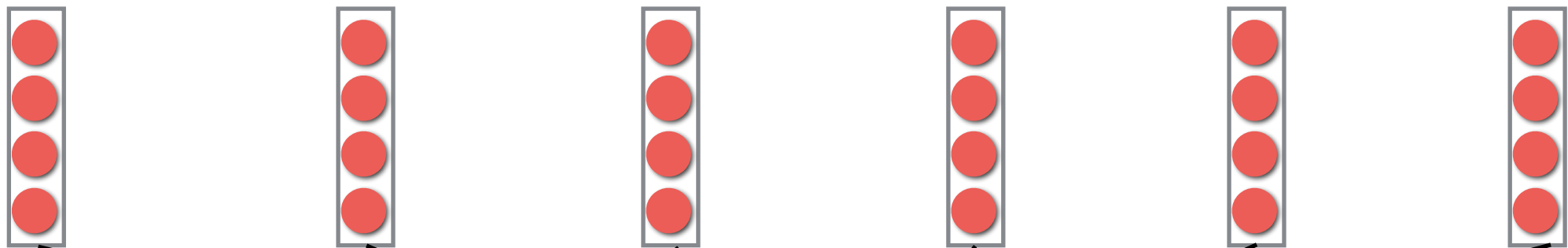
prediction

Recurrent NN,
ConvNet,
Recursive NN,
...

Learning language

Memorize

John gave the book to Mary



prediction

Recurrent NN,
ConvNet,
Recursive NN,
...

Generalize

Learning language

CHALLENGE 1: IDIOMS

CHALLENGE 2: MORPHOLOGY

Learning language

CHALLENGE 1: IDIOMS

John saw the football

SEE(JOHN, FOOTBALL)

John saw the bucket

SEE(JOHN, BUCKET)

CHALLENGE 2: MORPHOLOGY

Learning language

CHALLENGE 1: IDIOMS

John saw the football

SEE(JOHN, FOOTBALL)

John saw the bucket

SEE(JOHN, BUCKET)

John kicked the football

KICK(JOHN, FOOTBALL)

John kicked the bucket

DIE(JOHN)

CHALLENGE 2: MORPHOLOGY

Learning language

CHALLENGE 1: IDIOMS

John saw the football

SEE(JOHN, FOOTBALL)

John saw the bucket

SEE(JOHN, BUCKET)

John kicked the football

KICK(JOHN, FOOTBALL)

John kicked the bucket

DIE(JOHN)

CHALLENGE 2: MORPHOLOGY

Learning language

CHALLENGE 1: IDIOMS

John saw the football

SEE(JOHN, FOOTBALL)

John saw the bucket

SEE(JOHN, BUCKET)

John kicked the football

KICK(JOHN, FOOTBALL)

John kicked the bucket

DIE(JOHN)

CHALLENGE 2: MORPHOLOGY

cool | coooooo1 | coooooooooooooo1



Learning language

CHALLENGE 1: IDIOMS

John saw the football

SEE(JOHN, FOOTBALL)

John saw the bucket

SEE(JOHN, BUCKET)

John kicked the football

KICK(JOHN, FOOTBALL)

John kicked the bucket

DIE(JOHN)

CHALLENGE 2: MORPHOLOGY

cool | cooooool | coooooooooool



cat + s = cats



Learning language

CHALLENGE 1: IDIOMS

John saw the football

SEE(JOHN, FOOTBALL)

John saw the bucket

SEE(JOHN, BUCKET)

John kicked the football

KICK(JOHN, FOOTBALL)

John kicked the bucket

DIE(JOHN)

CHALLENGE 2: MORPHOLOGY

cool | cooooool | coooooooooool



cat + s = cats



bat + s = bats



Learning language

CHALLENGE 1: IDIOMS

John saw the football

SEE(JOHN, FOOTBALL)

John saw the bucket

SEE(JOHN, BUCKET)

John kicked the football

KICK(JOHN, FOOTBALL)

John kicked the bucket

DIE(JOHN)

CHALLENGE 2: MORPHOLOGY

cool | cooooool | coooooooooool



cat + s = cats



bat + s = bats



Compositional words

cats

Memorize

Generalize

Compositional words

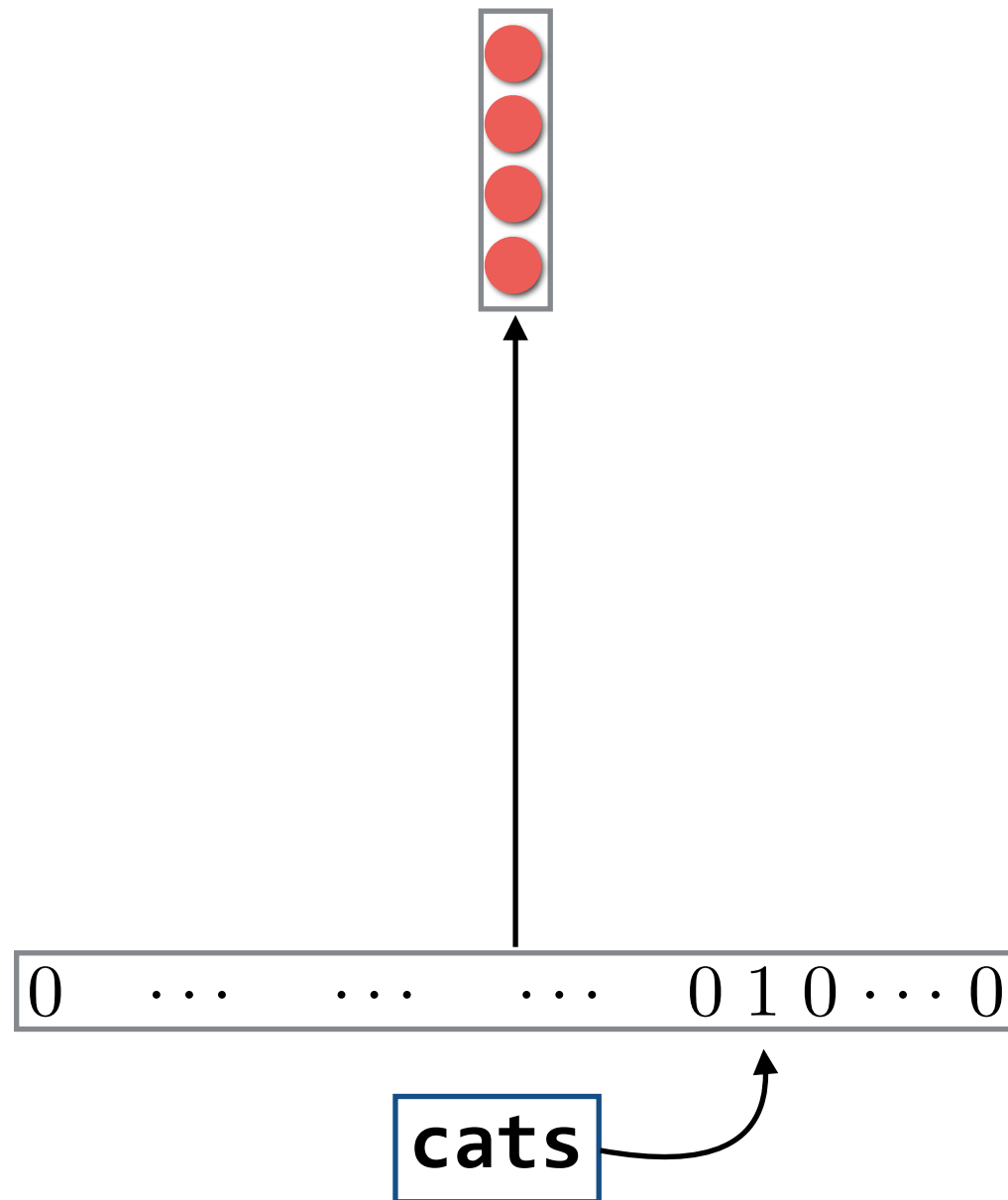
0 0 1 0 ... 0

cats

Memorize

Generalize

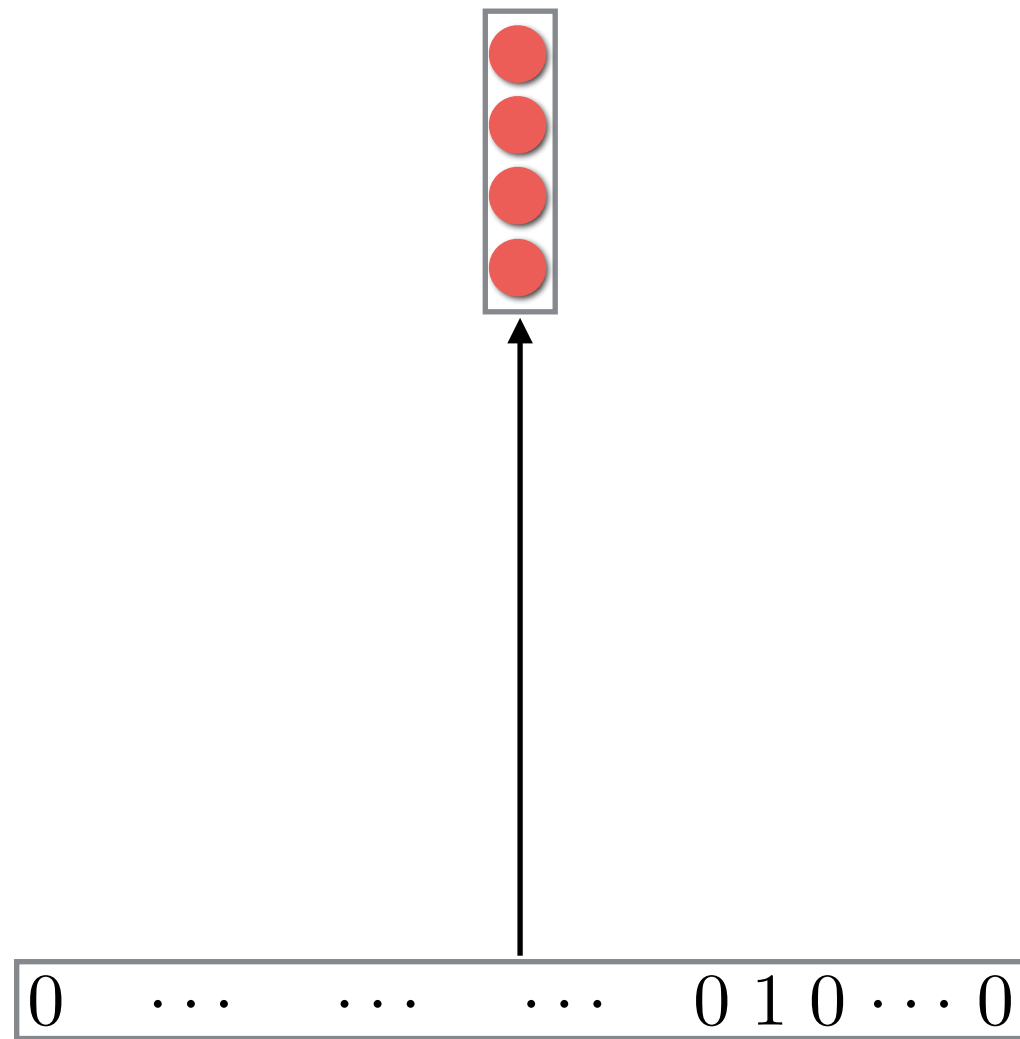
Compositional words



Memorize

Generalize

Compositional words



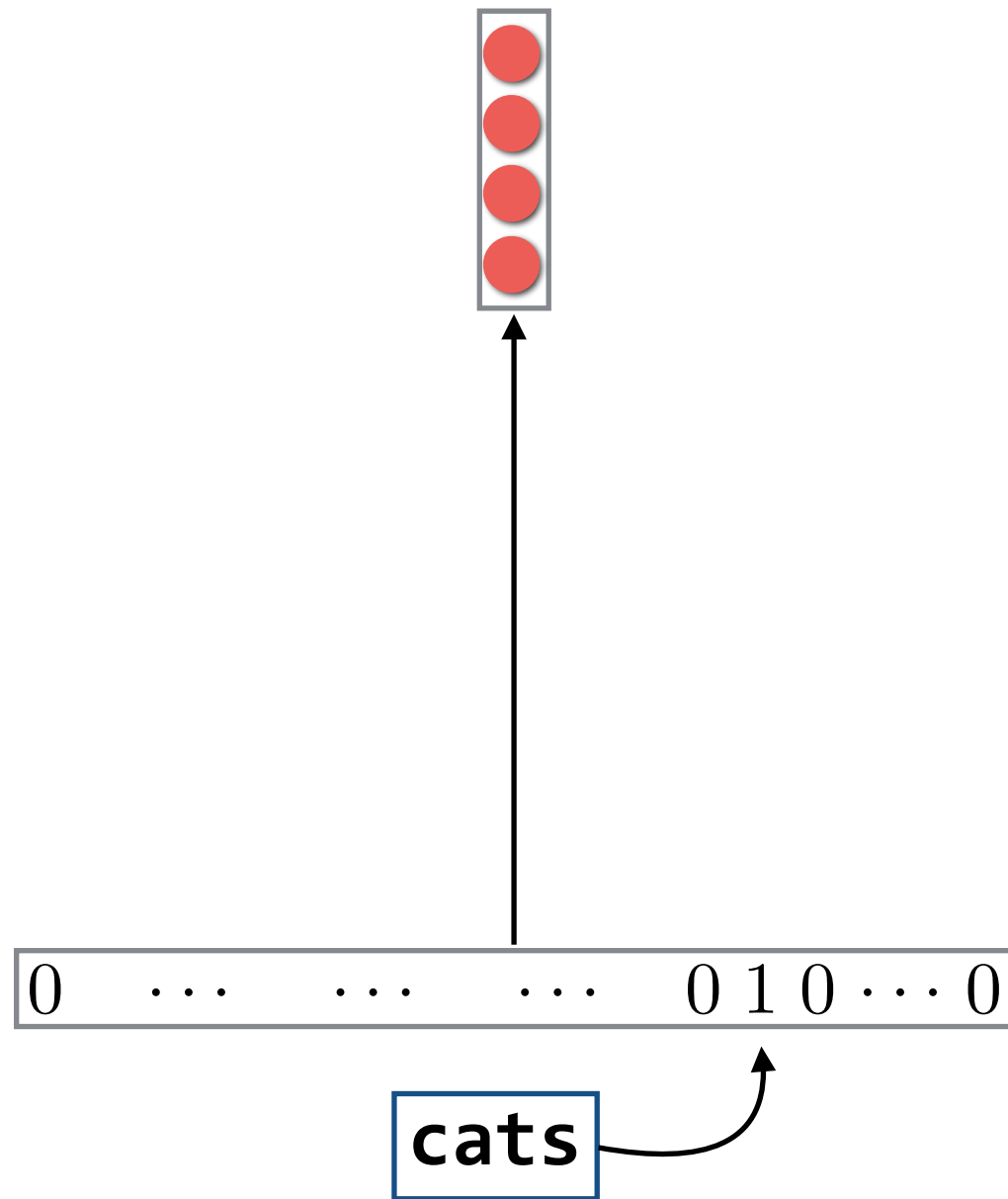
cats

Memorize

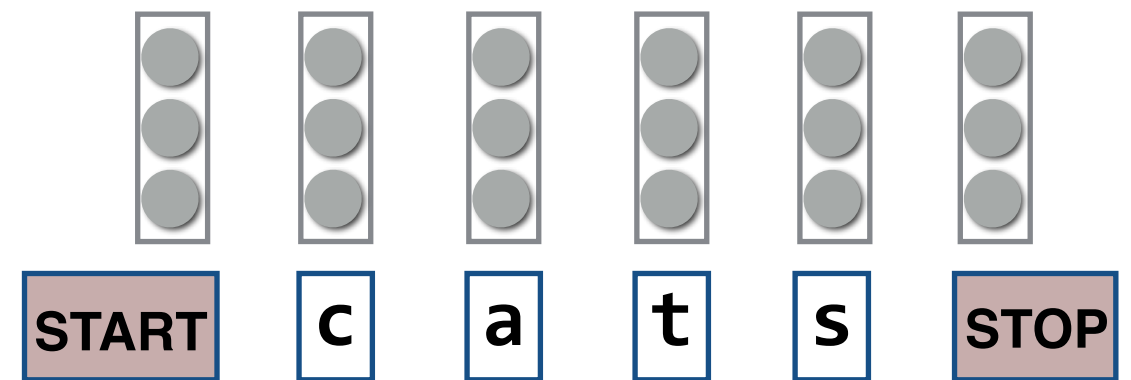
START c a t s STOP

Generalize

Compositional words

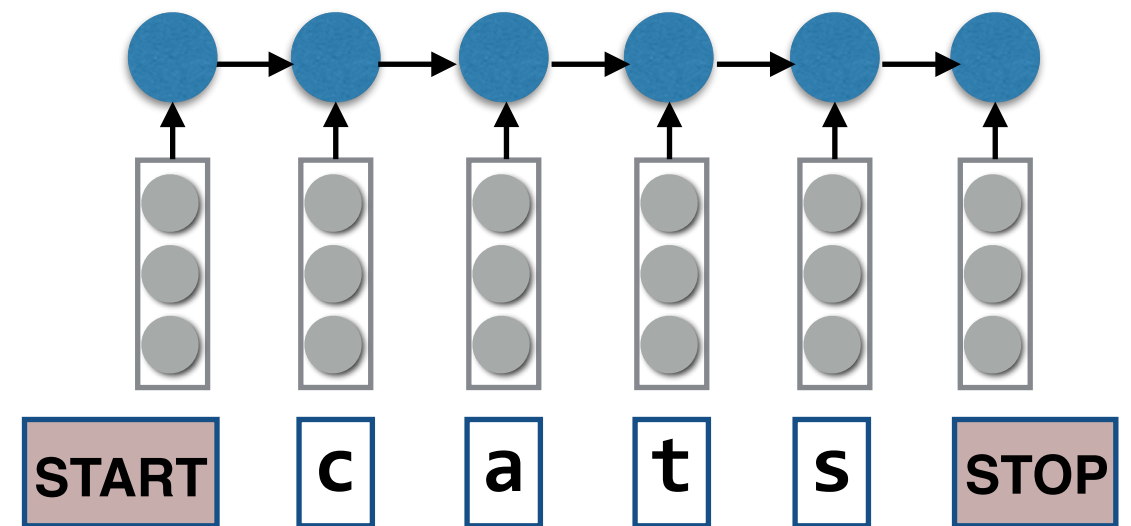
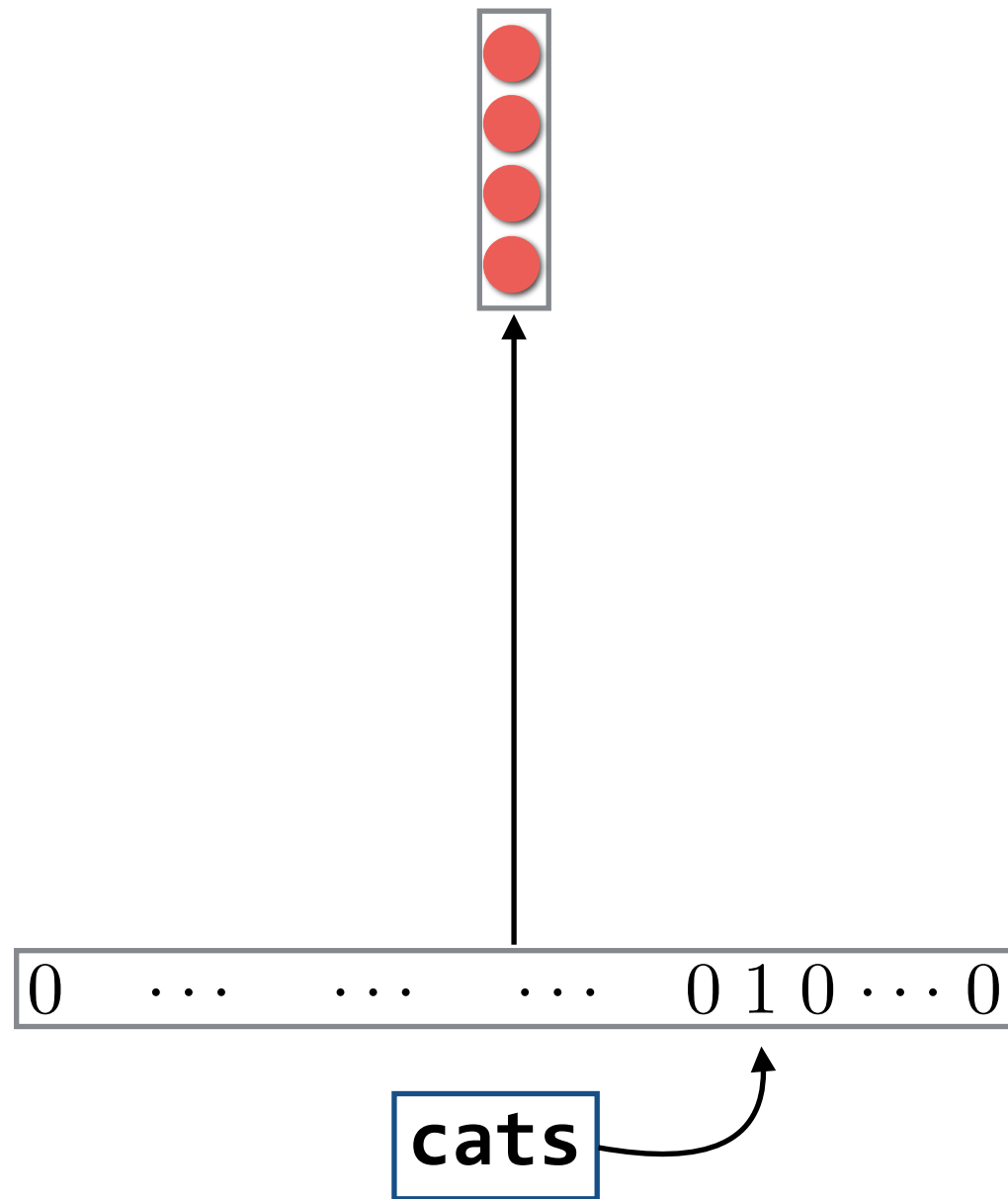


Memorize

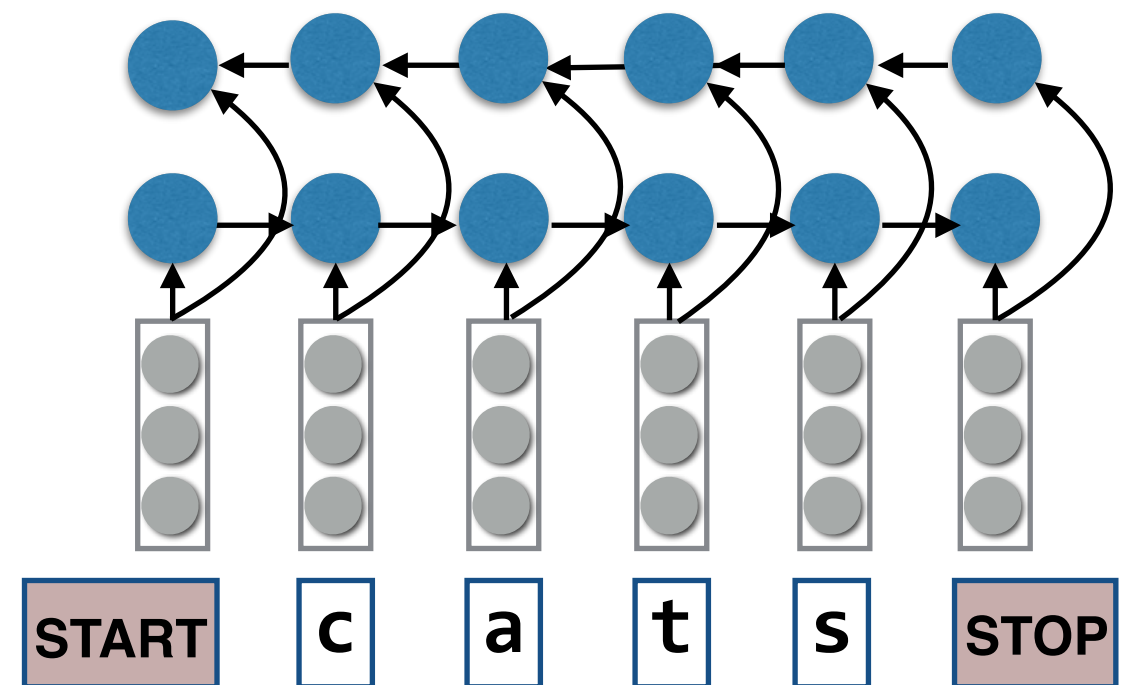
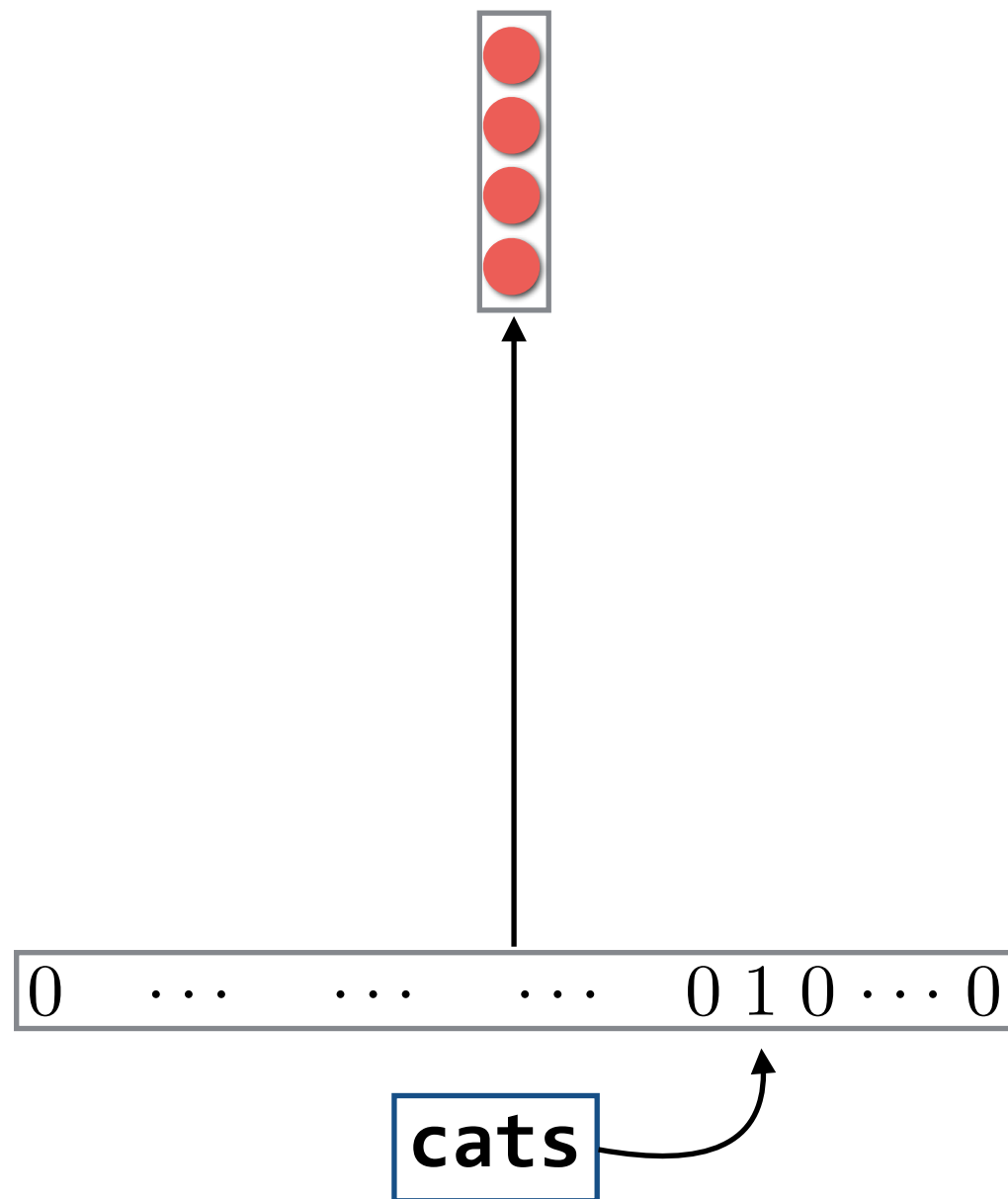


Generalize

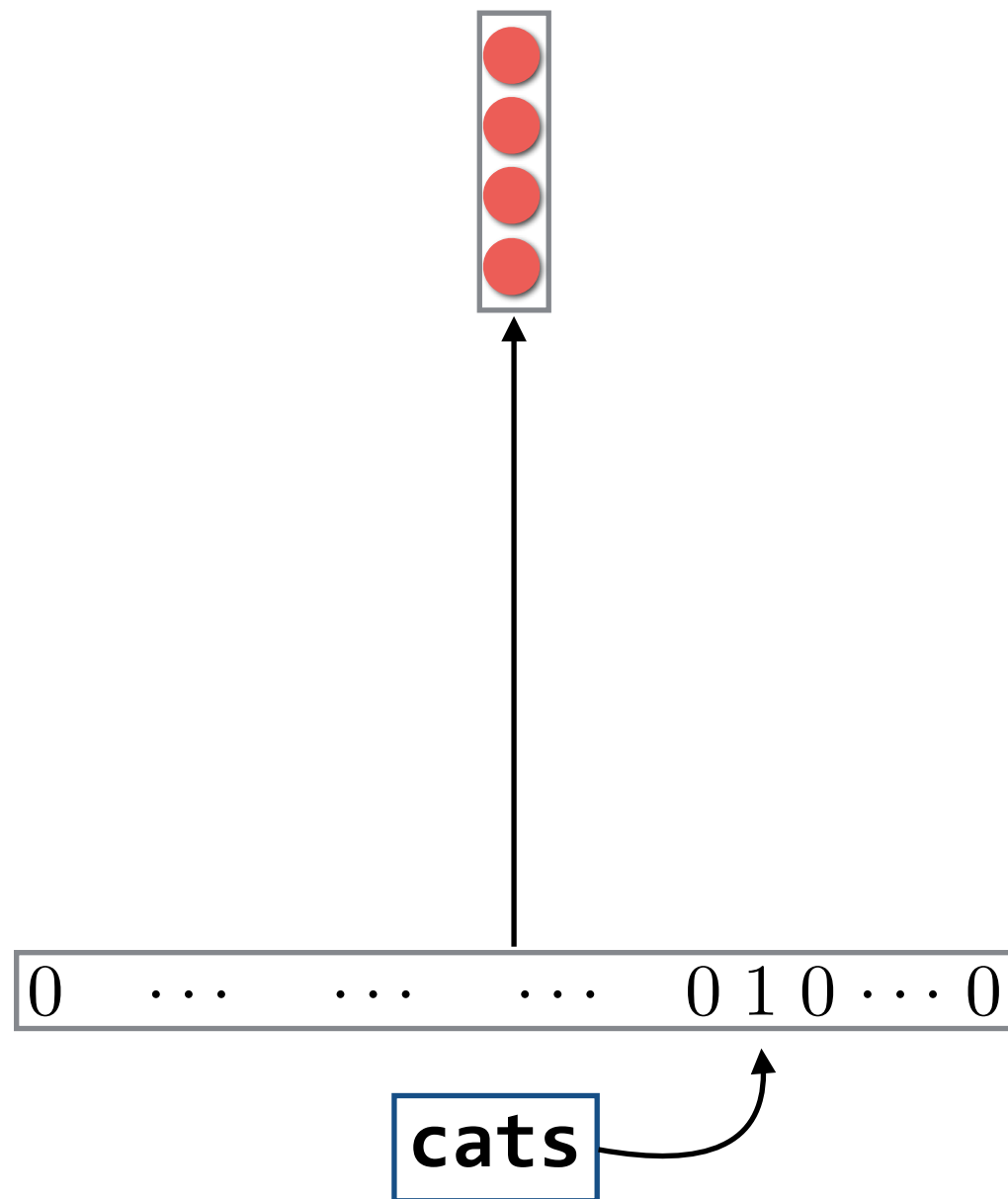
Compositional words



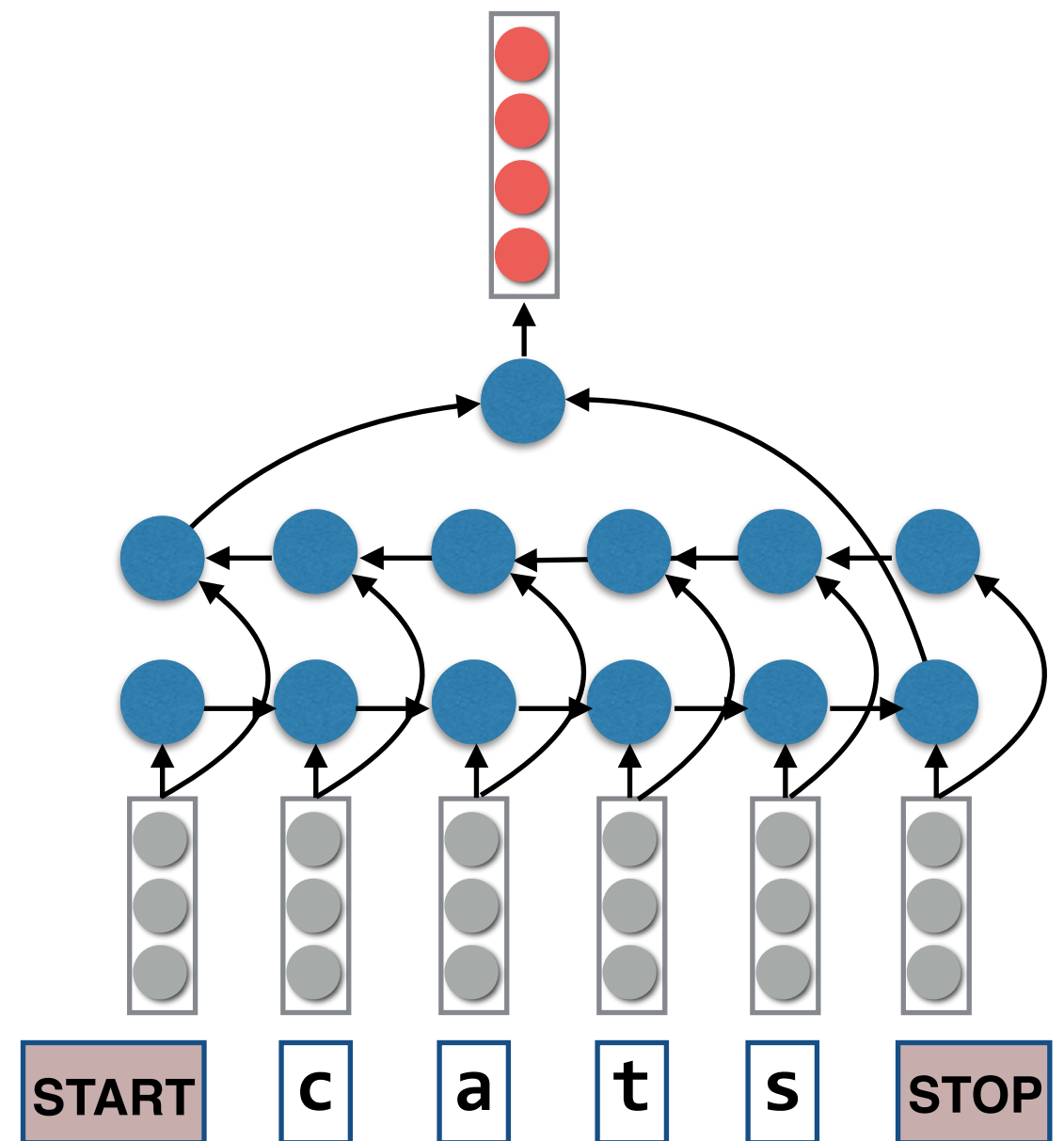
Compositional words



Compositional words



Memorize



Generalize

Compositional words

Questions

- Does a “compositional” model have the capacity to learn the “arbitrariness” that is required?
- We might think so—RNNs/LSTMs can definitely overfit!
- Will we see better improvements in languages with more going on in the morphology?

Example

Dependency parsing

I saw her duck

Example

Dependency parsing

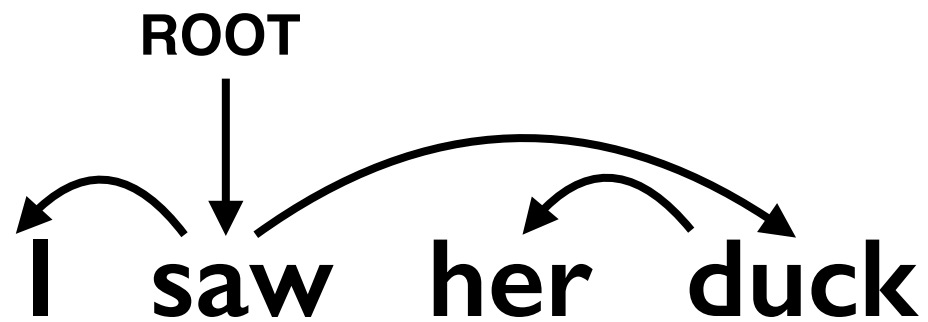


I saw her duck



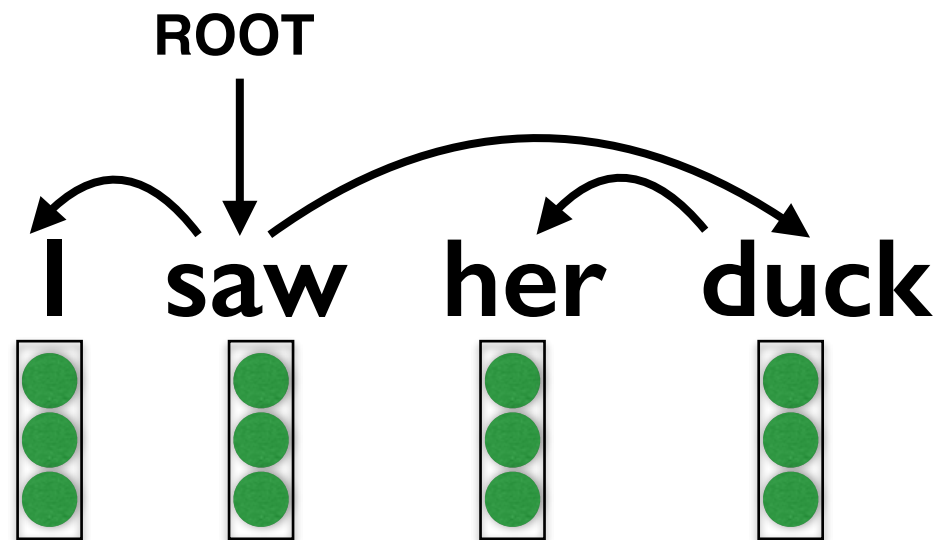
Example

Dependency parsing



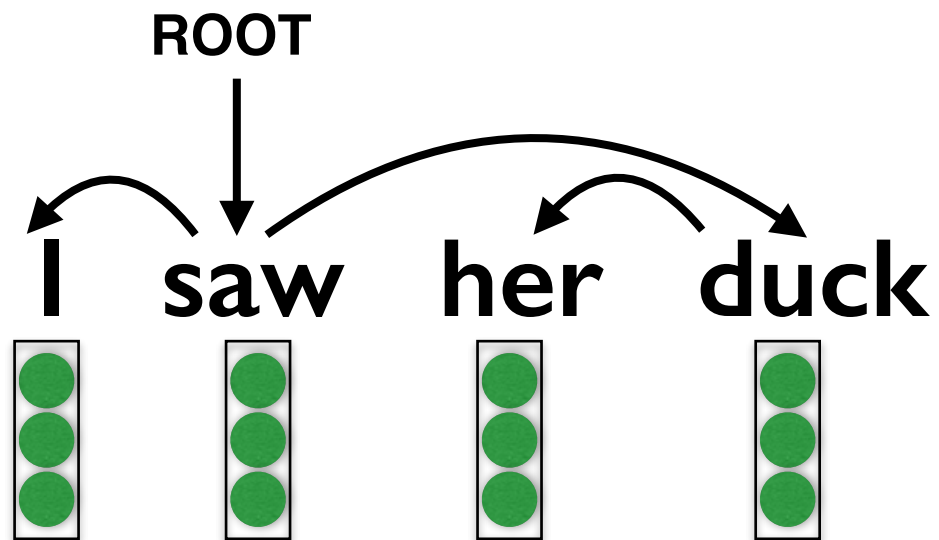
Example

Dependency parsing

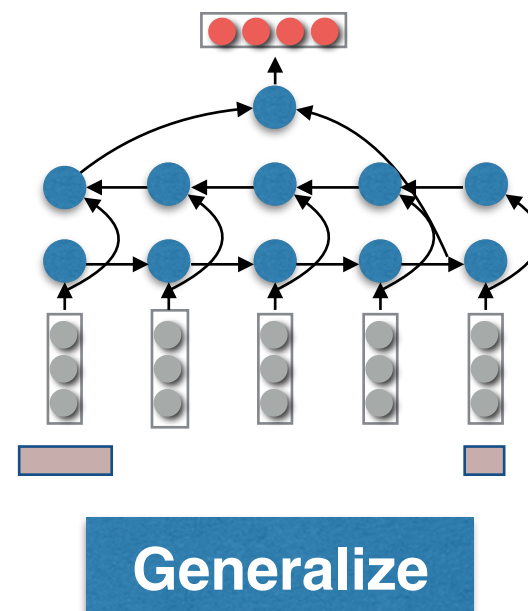
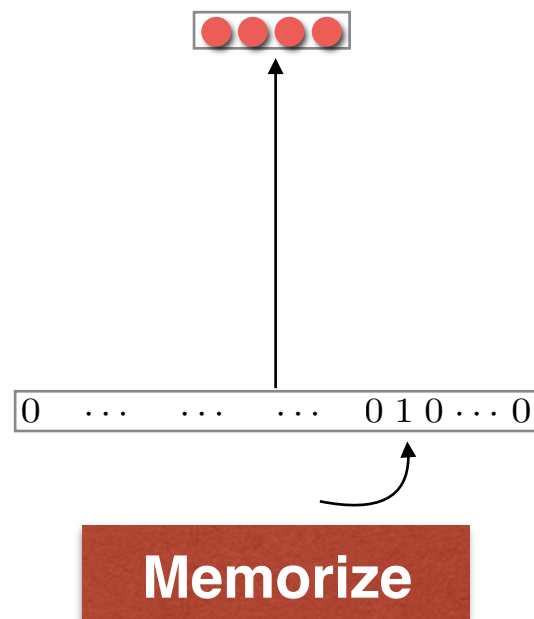


Example

Dependency parsing



Word embedding models:



Dependency parsing

CharLSTM > Word Lookup

	Word	Chars	Δ
--	------	-------	----------

English	91.2	91.5	+0.3
---------	------	------	-------------

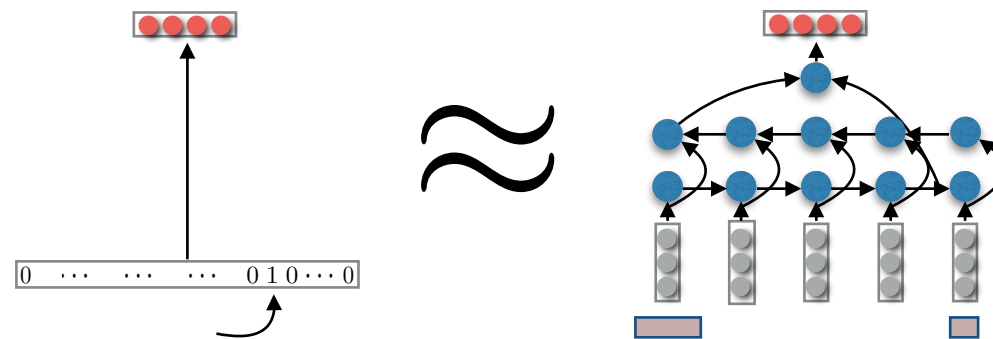
Dependency parsing

CharLSTM > Word Lookup

	Word	Chars	Δ
--	------	-------	----------

English	91.2	91.5	+0.3
---------	------	------	-------------

In English parsing, the character LSTM is roughly equivalent to the lookup approach.

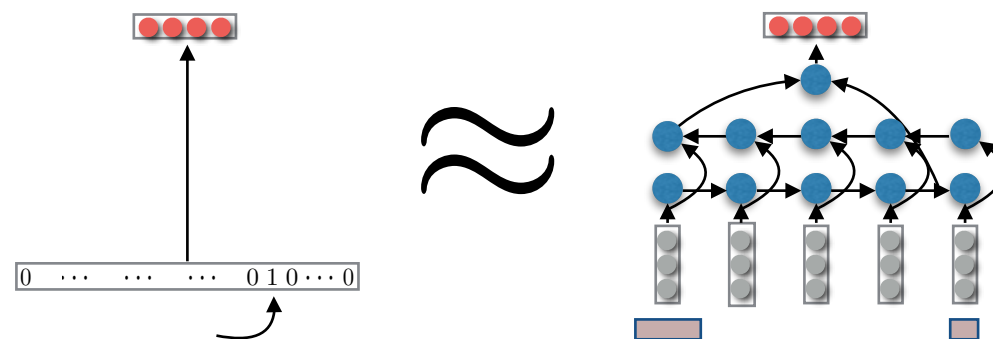


Dependency parsing

CharLSTM > Word Lookup

	Word	Chars	Δ
English	91.2	91.5	+0.3

In English parsing, the character LSTM is roughly equivalent to the lookup approach.



What about languages with richer lexicons?

Turkish: *Muvaffakiyetsizleştiricileştiriveremeyebileceklerimizdenmişsinizcesine*

Hungarian: *Megszentségteleníthetetleniségeskedéseitekért*

Dependency parsing

CharLSTM > Word Lookup

	Word	Chars	Δ
--	------	-------	----------

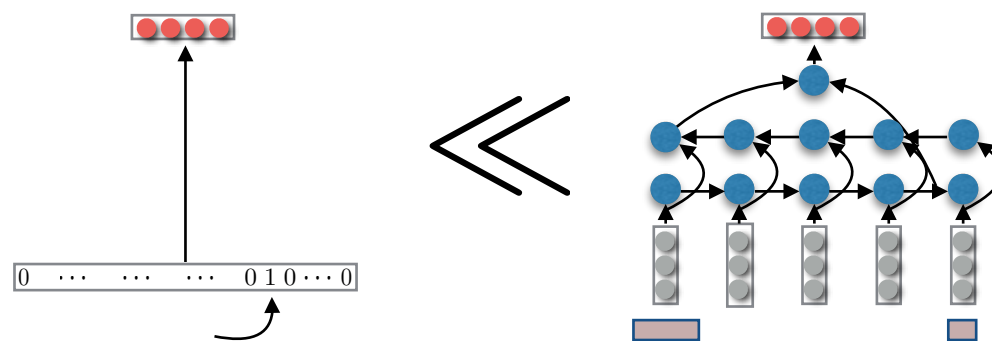
English	91.2	91.5	+0.3
---------	------	------	------

Dependency parsing

CharLSTM > Word Lookup

	Word	Chars	Δ
English	91.2	91.5	+0.3
Turkish	71.7	76.3	+4.6
Hungarian	72.8	80.4	+7.6
Basque	77.1	85.2	+8.1
Korean	78.7	88.4	+9.7

In **agglutinative** languages,

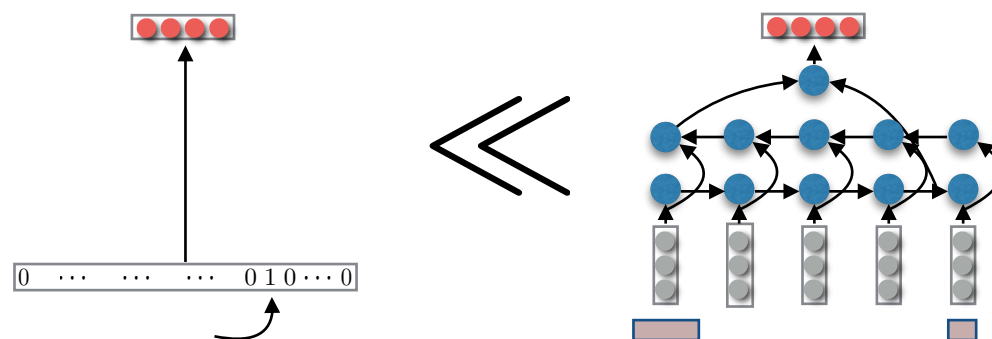


Dependency parsing

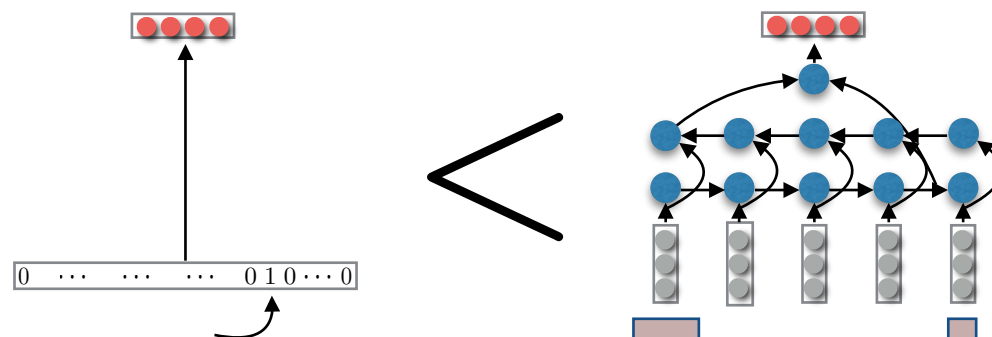
CharLSTM > Word Lookup

	Word	Chars	Δ
English	91.2	91.5	+0.3
Turkish	71.7	76.3	+4.6
Hungarian	72.8	80.4	+7.6
Basque	77.1	85.2	+8.1
Korean	78.7	88.4	+9.7
Swedish	76.4	79.2	+3.2
Swedish	76.4	79.2	+2.8
Arabic	85.2	86.1	+0.9

In **agglutinative** languages,



In **fusional/templatic** languages,

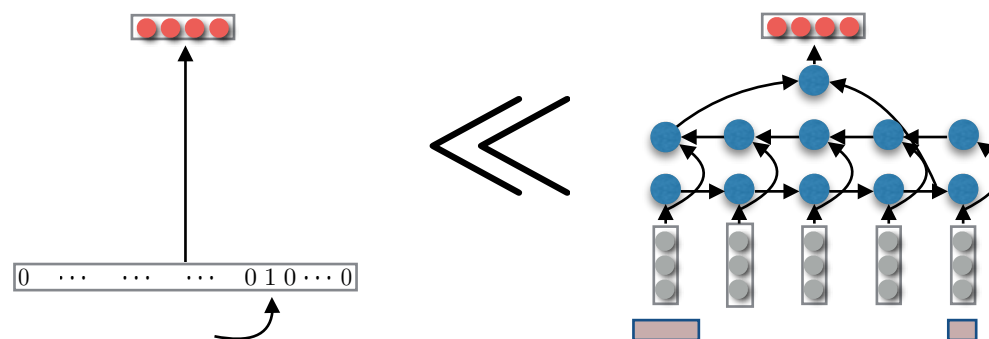


Dependency parsing

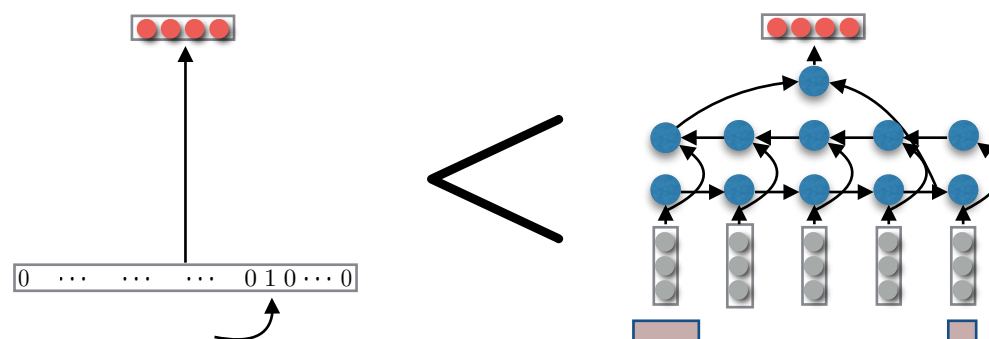
CharLSTM > Word Lookup

	Word	Chars	Δ
English	91.2	91.5	+0.3
Turkish	71.7	76.3	+4.6
Hungarian	72.8	80.4	+7.6
Basque	77.1	85.2	+8.1
Korean	78.7	88.4	+9.7
Swedish	76.4	79.2	+3.2
Swedish	76.4	79.2	+2.8
Arabic	85.2	86.1	+0.9
Chinese	79.1	79.9	+0.8

In **agglutinative** languages,



In **fusional/templatic** languages,



In **analytic** languages, the models are roughly equivalent.

Language modeling

Word similarities

query

increased **John**

Language modeling

Word similarities

query	increased	John
5 nearest neighbors	reduced	Richard
	improved	George
	expected	James
	decreased	Robert
	targeted	Edward

Language modeling

Word similarities

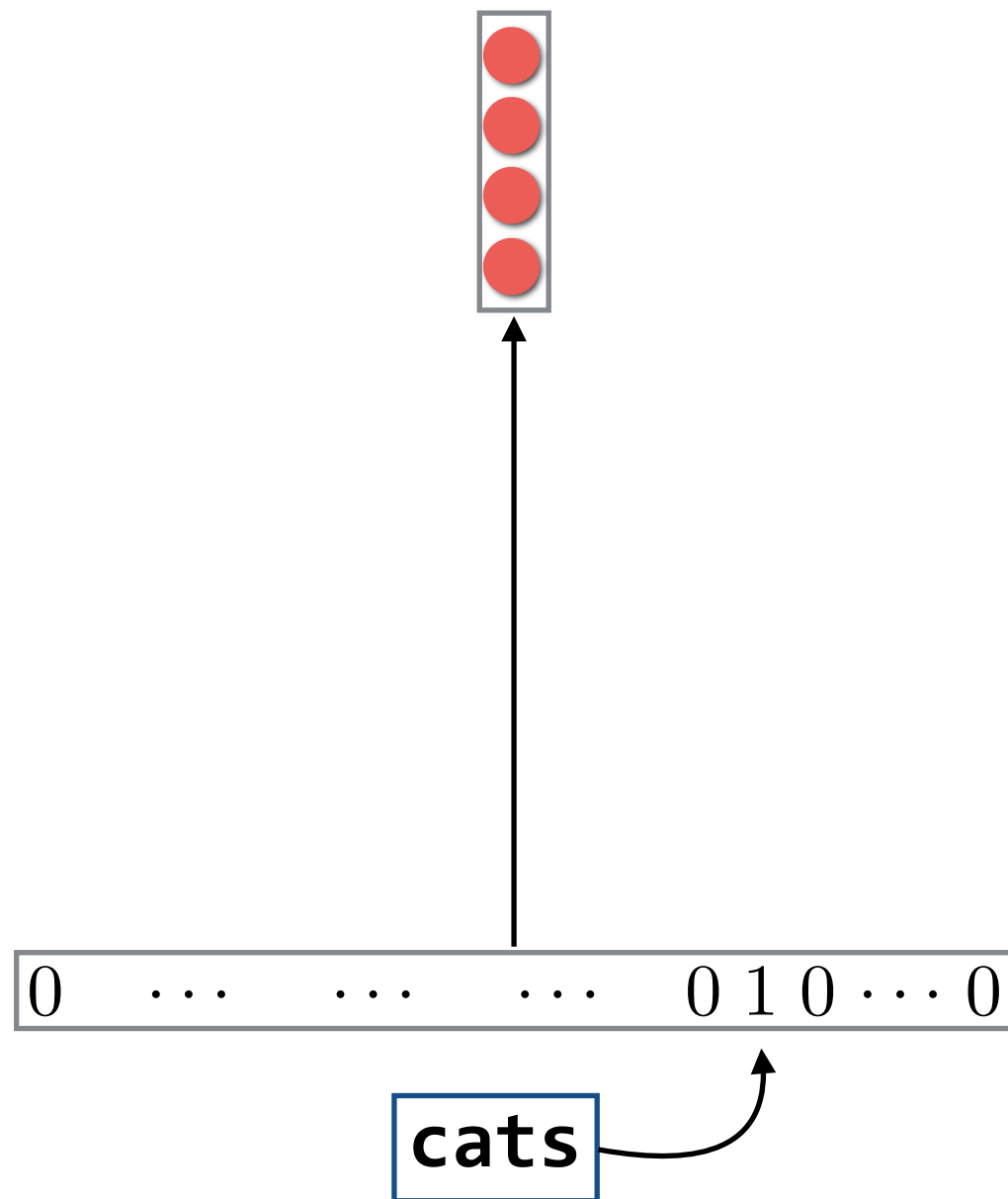
5 nearest neighbors	query	increased	John	Noahshire	phding
		reduced	Richard	Nottinghamshire	mixing
		improved	George	Bucharest	modelling
		expected	James	Saxony	styling
		decreased	Robert	Johannesburg	blaming
		targeted	Edward	Gloucestershire	christening

Character vs. word modeling

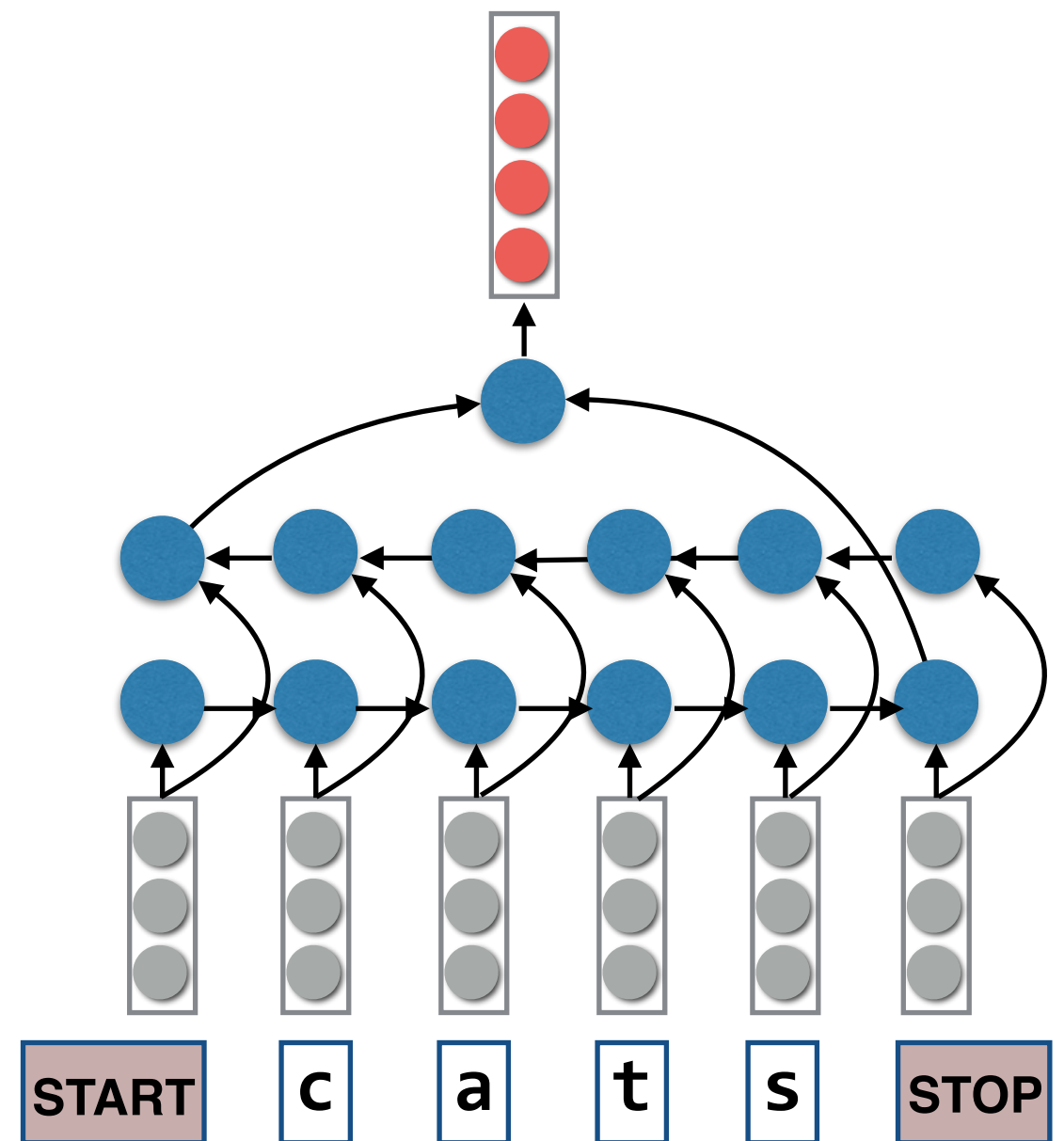
Summary

- Lots of exciting work from a variety of places
 - Google Brain: language models
 - Harvard/NYU (Kim, Rush, Sontag): language models
 - NYU/FB: document representation “from scratch”
 - CMU (me, Cohen, Salakhutdinov): Twitter, morphologically rich languages, translation
- Now for something a bit more controversial...

Structure-aware words



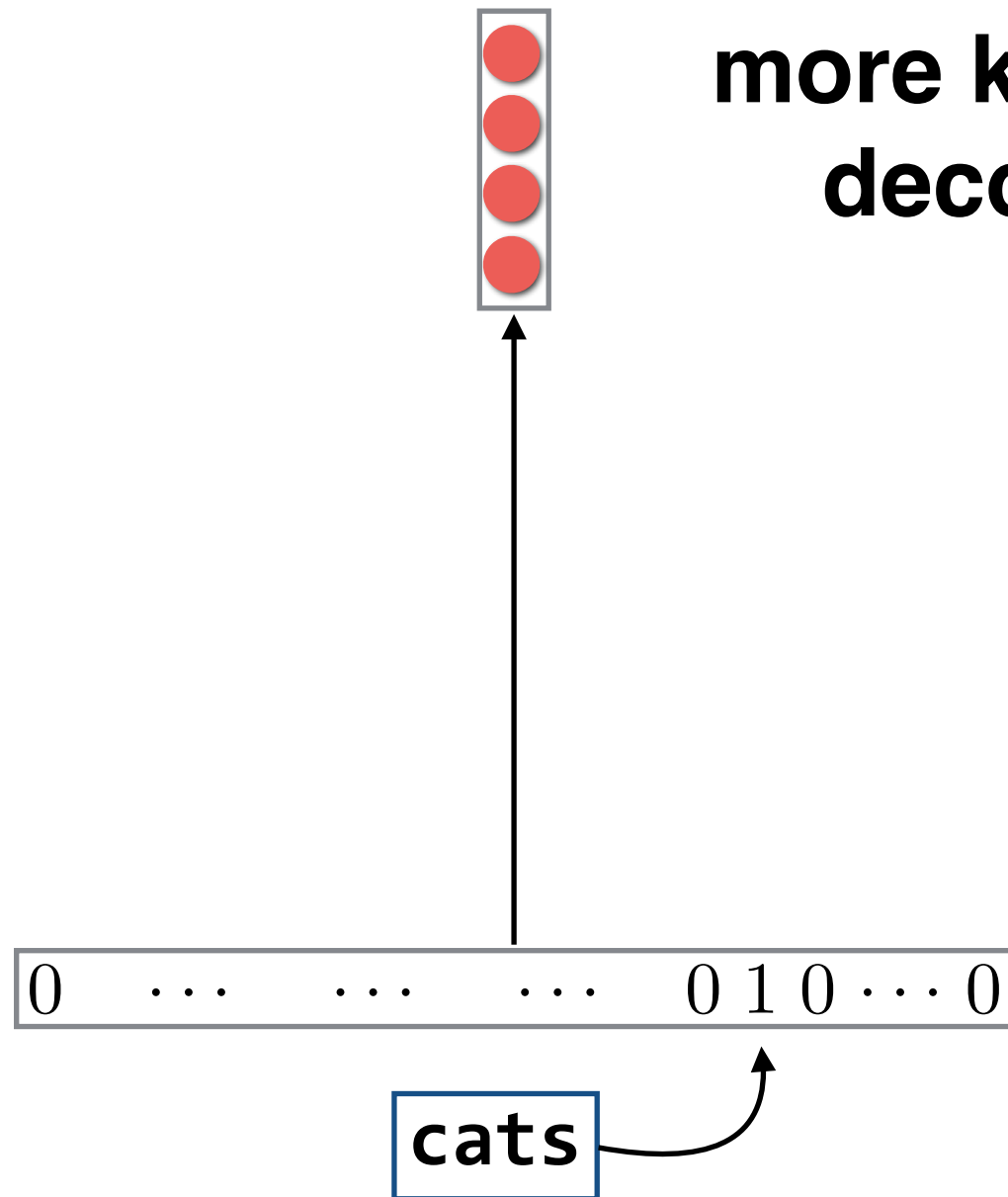
Memorize



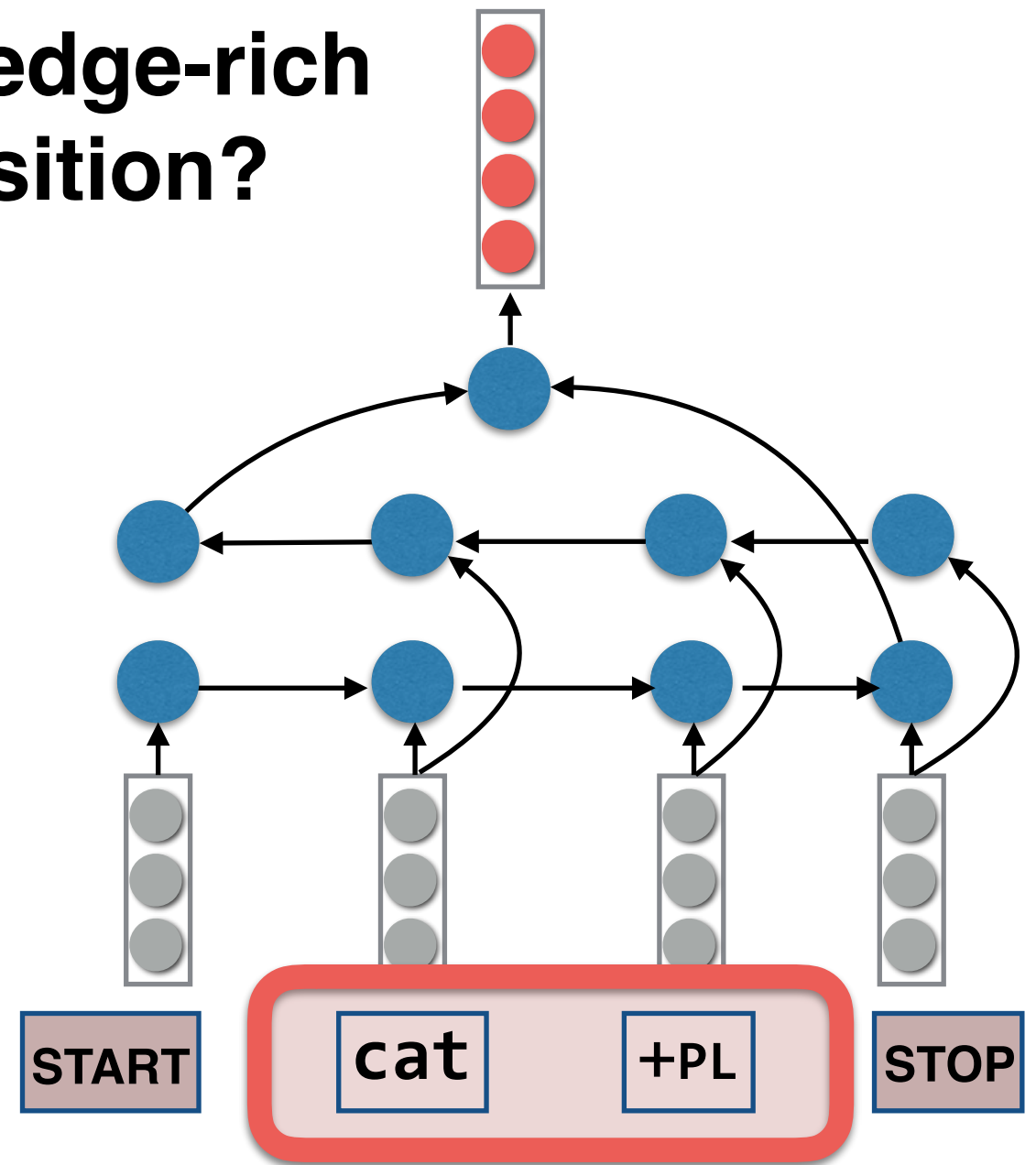
Generalize

Structure-aware words

Would we benefit from a more knowledge-rich decomposition?



Memorize



Generalize

Open Vocabulary LMs

- Rather than assuming a fixed vocabulary, model any sequence in Σ^* where Σ is the inventory of characters.

Open Vocabulary LMs

Turkish

Kosova

,

tekrar

eden

şikayetler

ışığında

özelleştirme

sürecini

incelemeye

alıyor

.

Open Vocabulary LMs

Turkish morphology

Morphological Analysis

Kosova	Kosova+Noun+Prop+A3sg+Pnon+Nom	
,	,+Punc	
tekrar	tekrar+Adverb	tekrar+Noun+A3sg+Pnon+Nom
eden	et+Verb+Pos^DB+Adj+PresPart	ede+Noun+A3sg+P2sg+Nom
şikayetler	şikayet+Noun+A3pl+Pnon+Nom	
ışığında	ışık+Noun+A3sg+P3sg+Loc	ışık+Noun+A3sg+P2sg+Loc
özelleştirme	özel+Adj^DB+Verb+Become^DB+Verb+Caus+Pos^DB+Noun+Inf2+A3sg+Pnon+Nom	özel+Adj^DB+Verb+Become^DB+Verb+Caus+Neg+Imp+A2sg
sürecini	süreç+Noun+A3sg+P3sg+Acc	süreç+Noun+A3sg+P2sg+Acc
incelemeye	incele+Verb+Pos^DB+Noun+Inf2+A3sg+Pnon+Dat	incel+Verb^DB+Verb+Able+Neg+Opt+A3sg
alıyor	al+Verb+Pos+Prog1+A3sg	
.	.+Punc	

Open Vocabulary LMs

Turkish morphology

Morphological Analysis

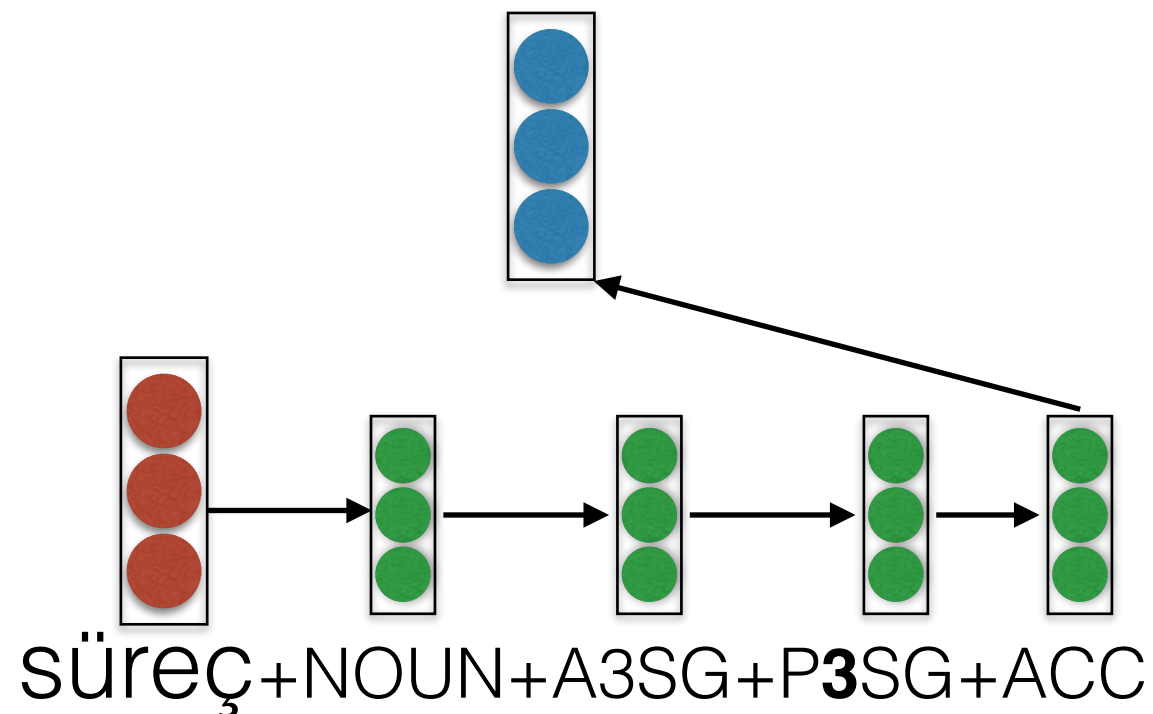
Kosova	Kosova+Noun+Prop+A3sg+Pnon+Nom	
,	,+Punc	
tekrar	tekrar+Adverb	tekrar+Noun+A3sg+Pnon+Nom
eden	et+Verb+Pos^DB+Adj+PresPart	ede+Noun+A3sg+P2sg+Nom
şikayetler	şikayet+Noun+A3pl+Pnon+Nom	
ışığında	ışık+Noun+A3sg+P3sg+Loc	ışık+Noun+A3sg+P2sg+Loc
özeleştirme	özel+Adi^DB+Verb+Become^DB+Verb+Caus+Pos^DB+Noun+Inf2+A3sg+Pnon+Nom	özel+Adi^DB+Verb+Become^DB+Verb+Caus+Neg+Imp+A2sg
sürecini	süreç+Noun+A3sg+P3sg+Acc	süreç+Noun+A3sg+P2sg+Acc
incelemeye	incele+verb+Pos^DB+Noun+Inf2+A3sg+Pnon+Dat	incele+verb^DB+verb+Able+Neg+Opt+A3sg
alıyor	al+Verb+Pos+Prog1+A3sg	
.	.+Punc	

Open Vocabulary LMs

Turkish morphology

Morphological Analysis

Kosova	Kosova+Noun+Prop+A3sg+Pnon+Nom	
,	,+Punc	
tekrar	tekrar+Adverb	tekrar+Noun+A3sg+Pnon+Nom
eden	et+Verb+Pos^DB+Adj+PresPart	ede+Noun+A3sg+P2sg+Nom
şikayetler	şikayet+Noun+A3pl+Pnon+Nom	
ışığında	ışık+Noun+A3sg+P3sg+Loc	ışık+Noun+A3sg+P2sg+Loc
özeleştirme	özel+Adi^DB+Verb+Become^DB+Verb+Caus+Pos^DB+Noun+Inf2+A3sg+Pnon+Nom	özel+Adi^DB+Verb+Become^DB+Verb+Caus+Neg+Imp+A2sg
sürecini	süreç+Noun+A3sg+P3sg+Acc	süreç+Noun+A3sg+P2sg+Acc
incelemeye	incele+verb+Pos^DB+Noun+Inf2+A3sg+Pnon+Dat	incele+verb^DB+verb+Able+Neg+Opt+A3sg
alıyor	al+Verb+Pos+Prog1+A3sg	
.	.+Punc	

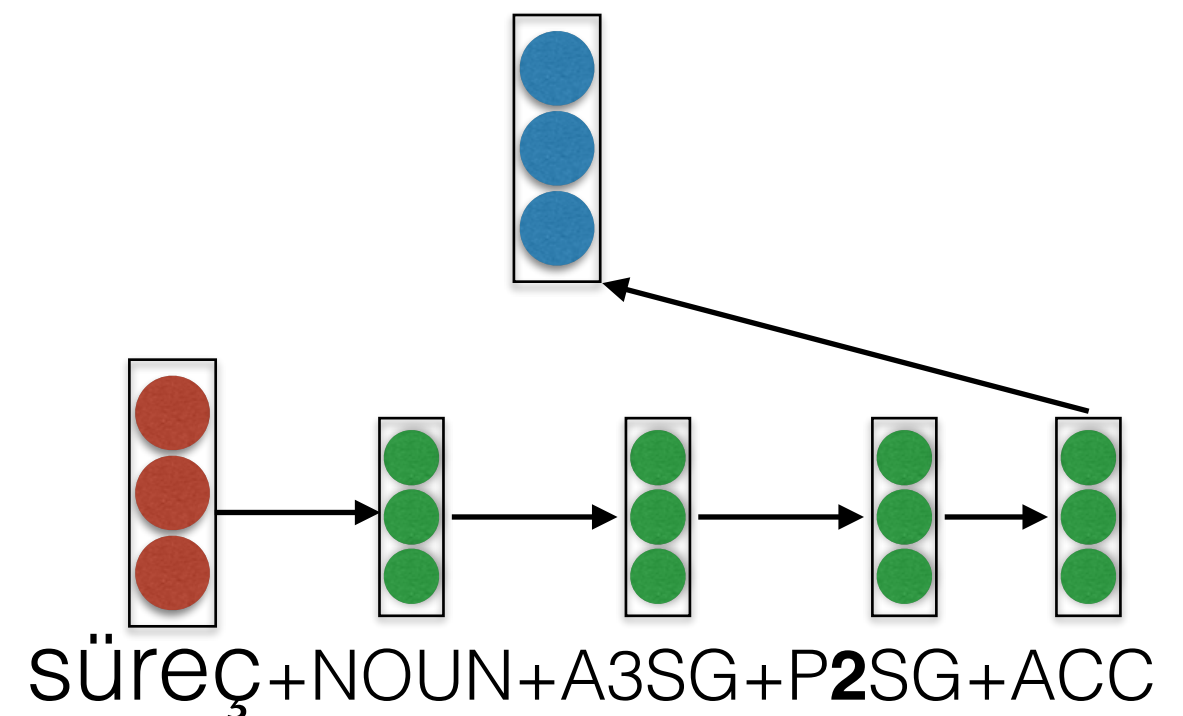
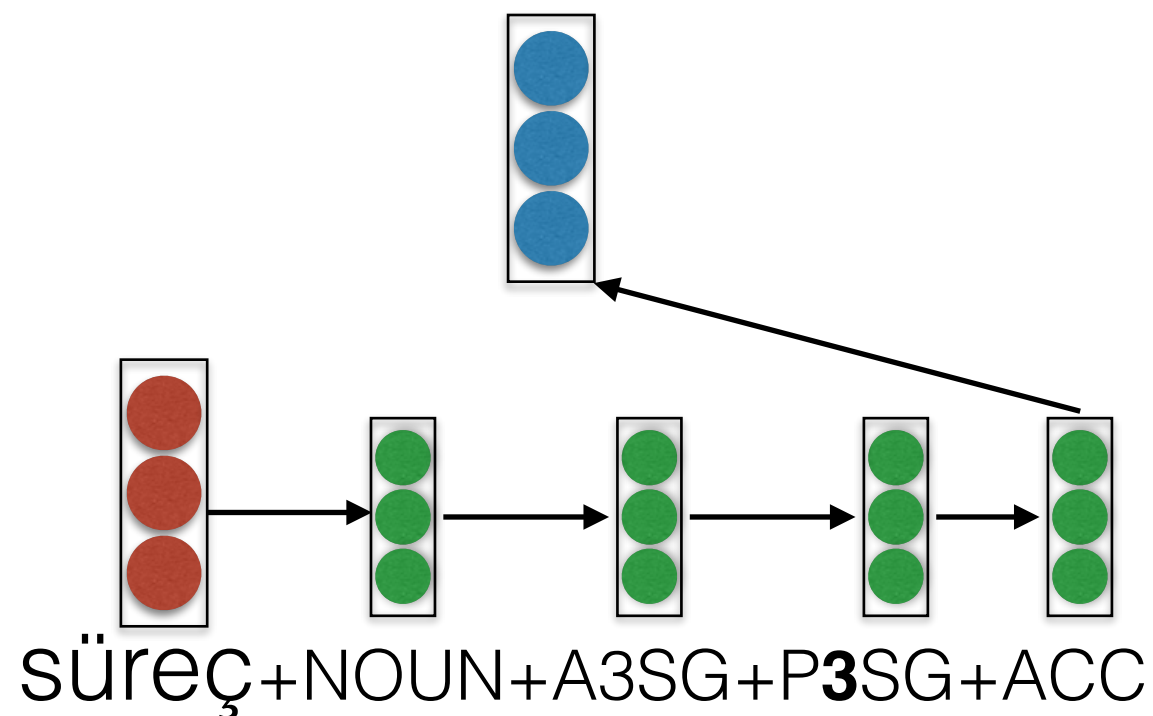


Open Vocabulary LMs

Turkish morphology

Morphological Analysis

Kosova	Kosova+Noun+Prop+A3sg+Pnon+Nom	
,	,+Punc	
tekrar	tekrar+Adverb	tekrar+Noun+A3sg+Pnon+Nom
eden	et+Verb+Pos^DB+Adj+PresPart	ede+Noun+A3sg+P2sg+Nom
şikayetler	şikayet+Noun+A3pl+Pnon+Nom	
ışığında	ışık+Noun+A3sg+P3sg+Loc	ışık+Noun+A3sg+P2sg+Loc
özeleştirme	özel+Adi^DB+Verb+Become^DB+Verb+Caus+Pos^DB+Noun+Inf2+A3sg+Pnon+Nom	özel+Adi^DB+Verb+Become^DB+Verb+Caus+Neg+Imp+A2sg
sürecini	süreç+Noun+A3sg+P3sg+Acc	süreç+Noun+A3sg+P2sg+Acc
incelemeye	incele+verb+Pos^DB+Noun+Inf2+A3sg+Pnon+Dat	incele+verb^DB+verb+Able+Neg+Opt+A3sg
alıyor	al+Verb+Pos+Prog1+A3sg	
.	.+Punc	

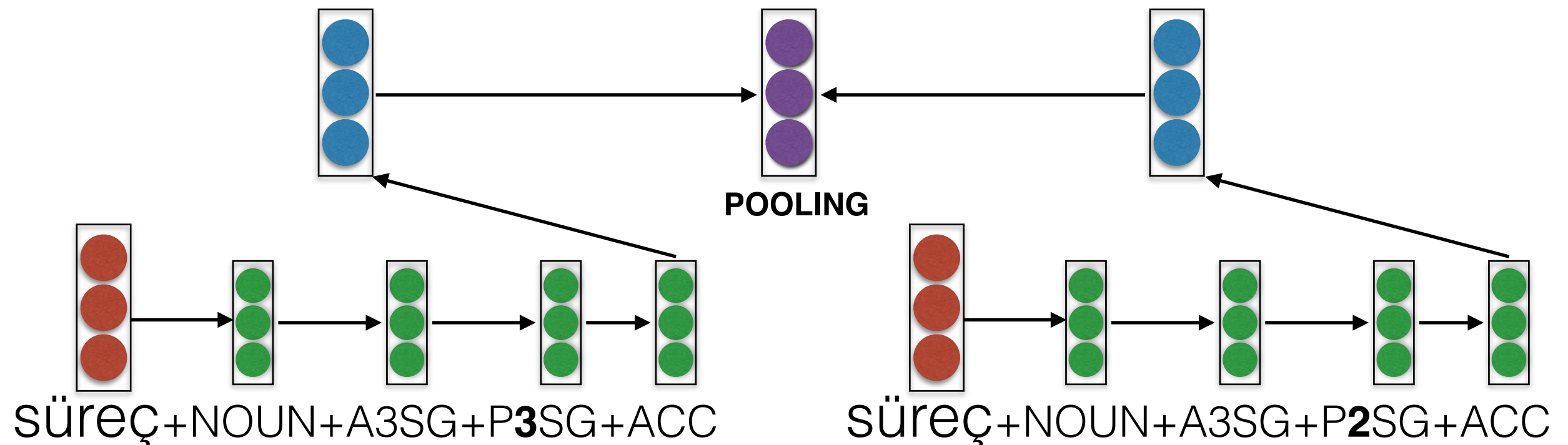


Open Vocabulary LMs

Turkish morphology

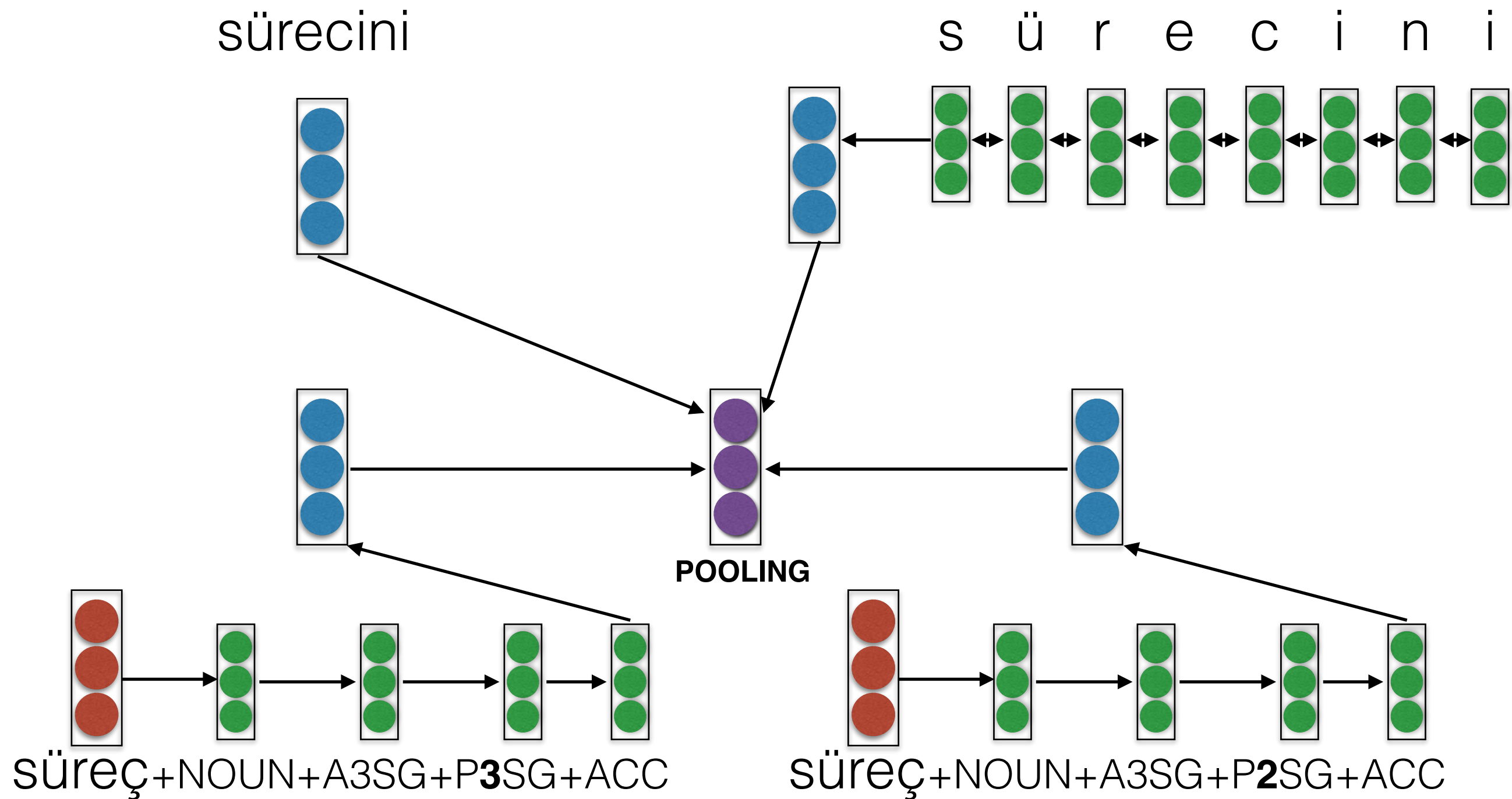
Morphological Analysis

Kosova	Kosova+Noun+Prop+A3sg+Pnon+Nom	
,	,+Punc	
tekrar	tekrar+Adverb	tekrar+Noun+A3sg+Pnon+Nom
eden	et+Verb+Pos^DB+Adj+PresPart	ede+Noun+A3sg+P2sg+Nom
şikayetler	şikayet+Noun+A3pl+Pnon+Nom	
ışığında	ışık+Noun+A3sg+P3sg+Loc	ışık+Noun+A3sg+P2sg+Loc
özeleştirme	özel+Adi^DB+Verb+Become^DB+Verb+Caus+Pos^DB+Noun+Inf2+A3sg+Pnon+Nom	özel+Adi^DB+Verb+Become^DB+Verb+Caus+Neg+Imp+A2sg
sürecini	süreç+Noun+A3sg+P3sg+Acc	süreç+Noun+A3sg+P2sg+Acc
incelemeye	incele+verb+Pos^DB+Noun+Inf2+A3sg+Pnon+Dat	incele+verb^DB+verb+Able+Neg+Opt+A3sg
alıyor	al+Verb+Pos+Prog1+A3sg	
.	.+Punc	

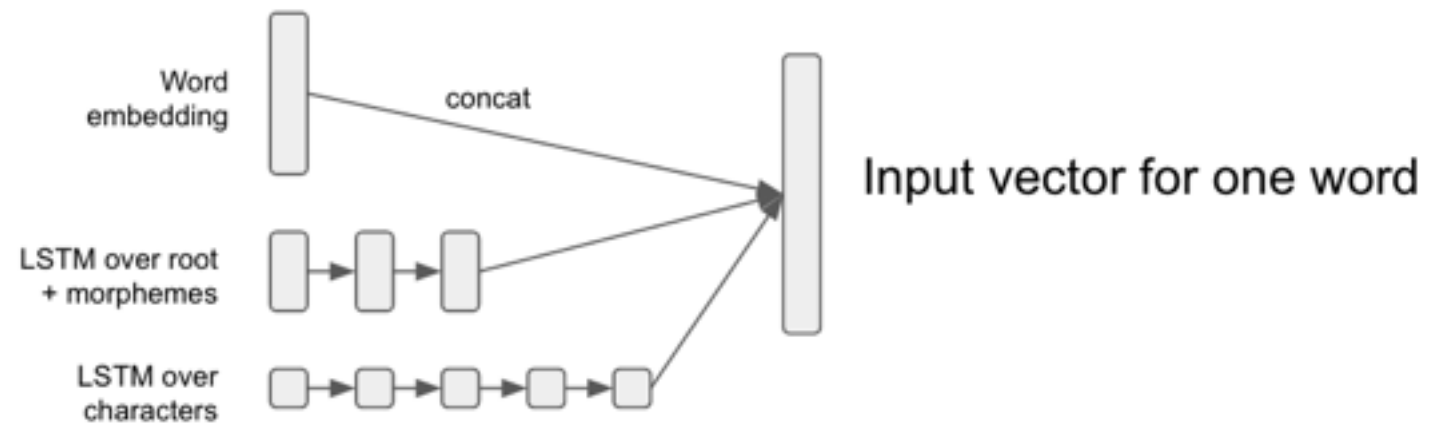


Open Vocabulary LMs

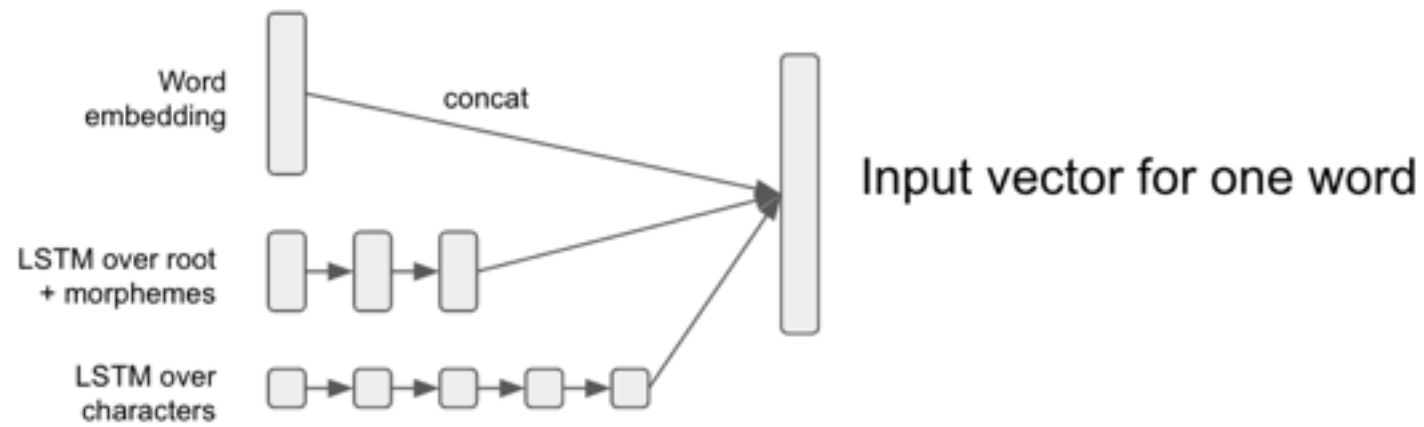
Turkish morphology



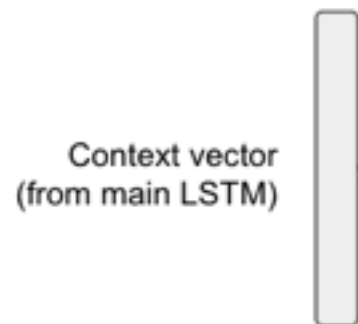
Input word representation:



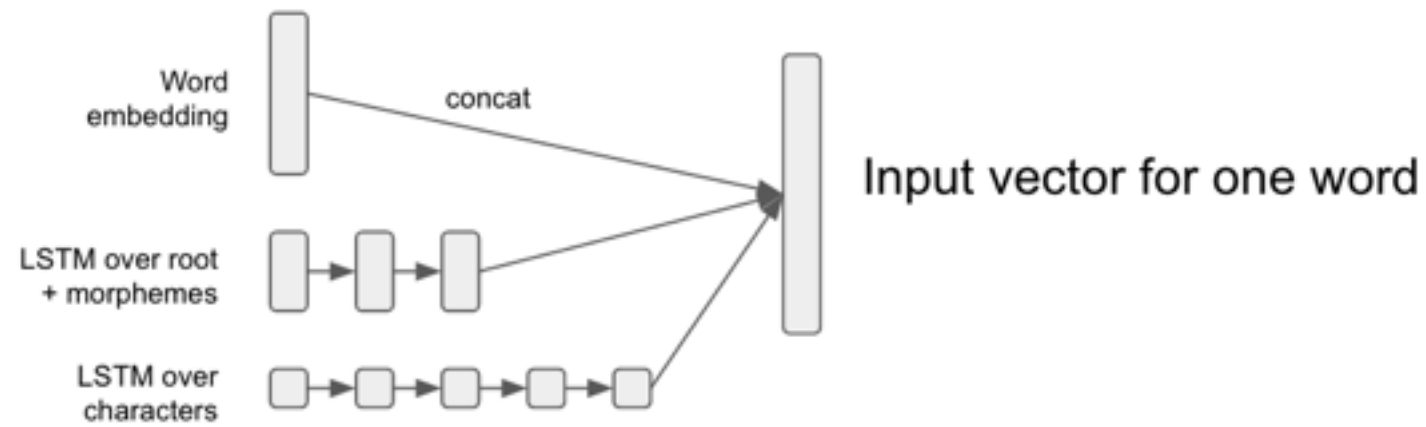
Input word representation:



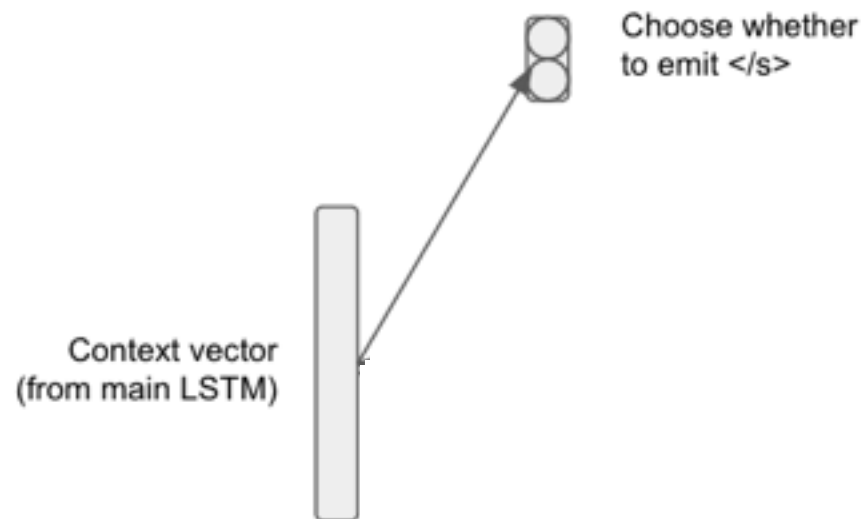
Output generation process: (marginalized)



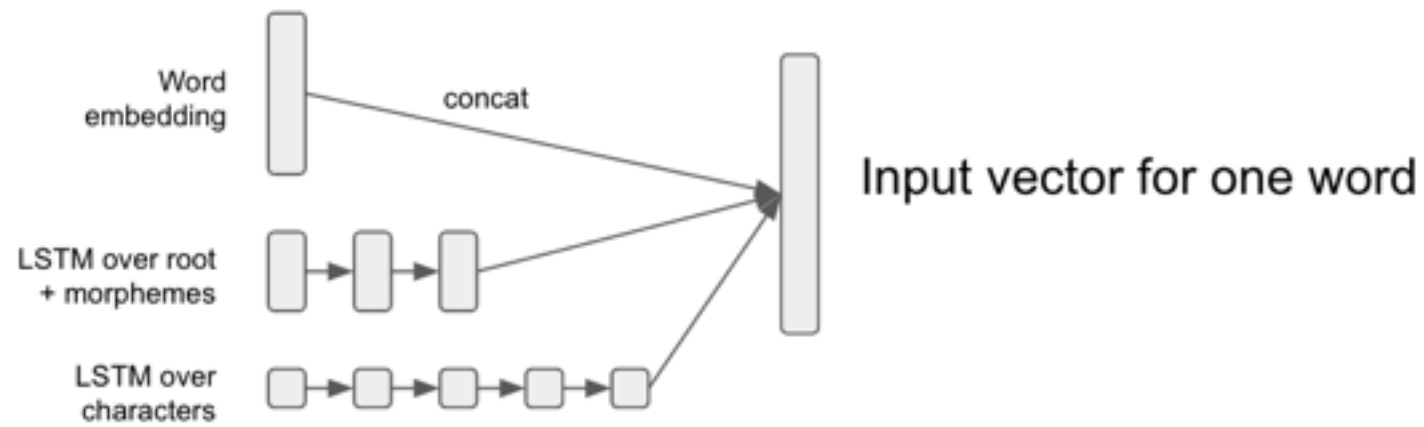
Input word representation:



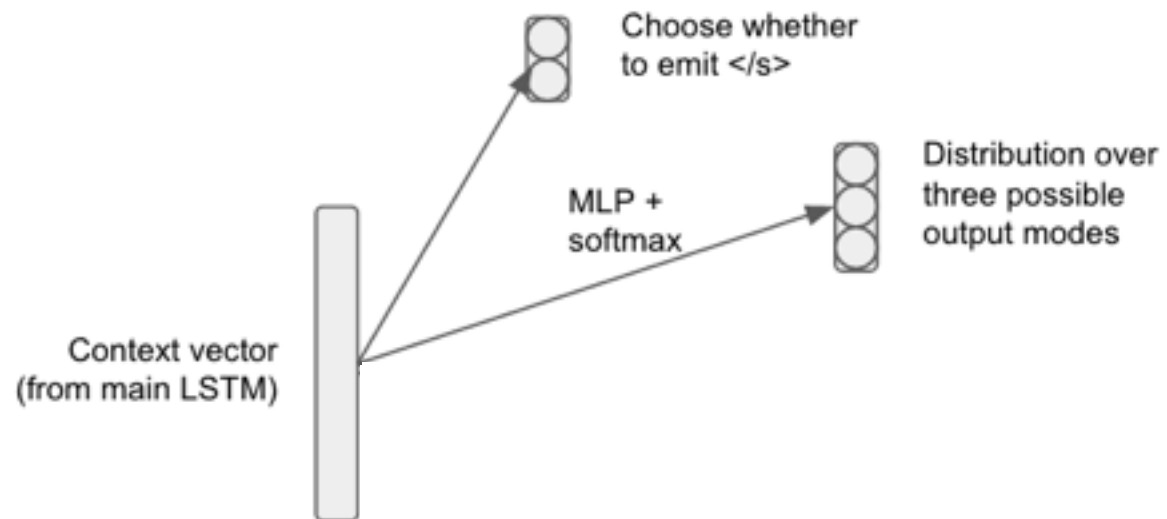
Output generation process: (marginalized)



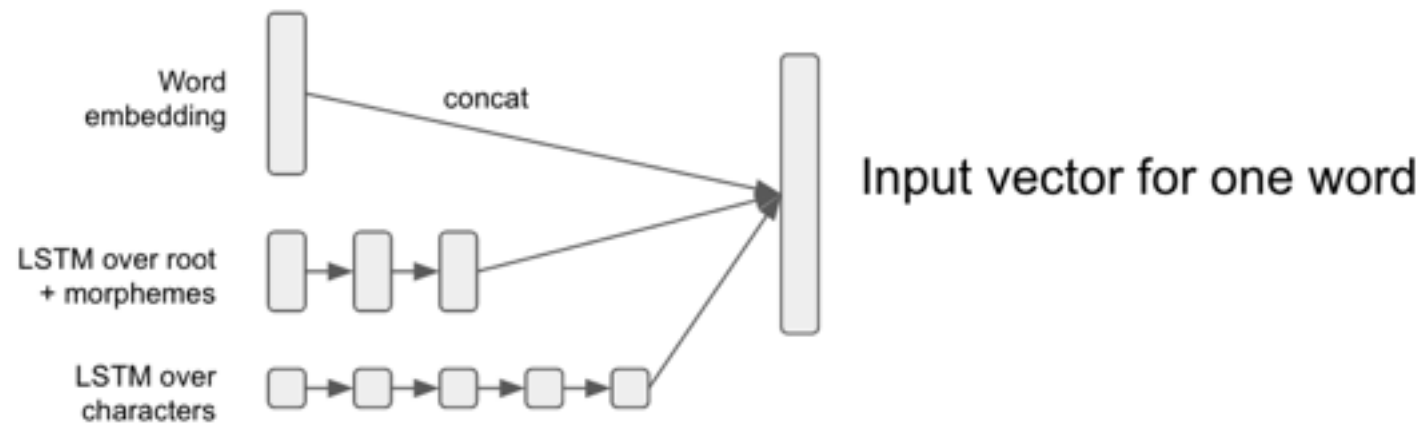
Input word representation:



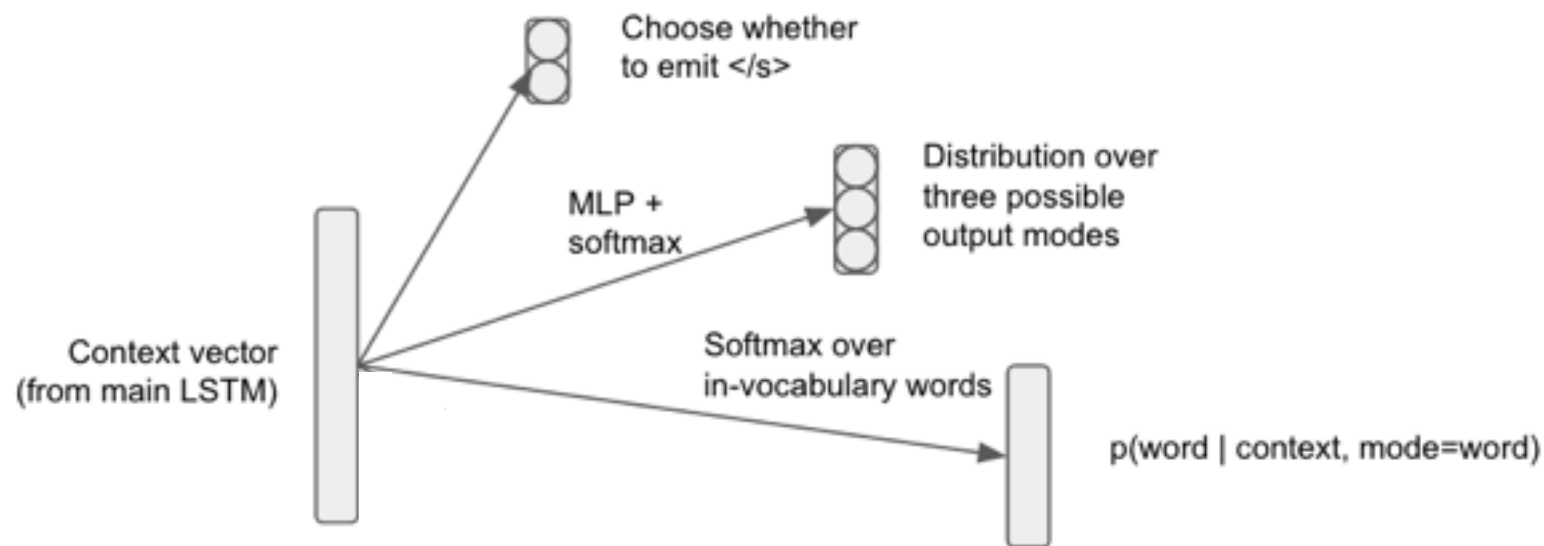
Output generation process: (marginalized)



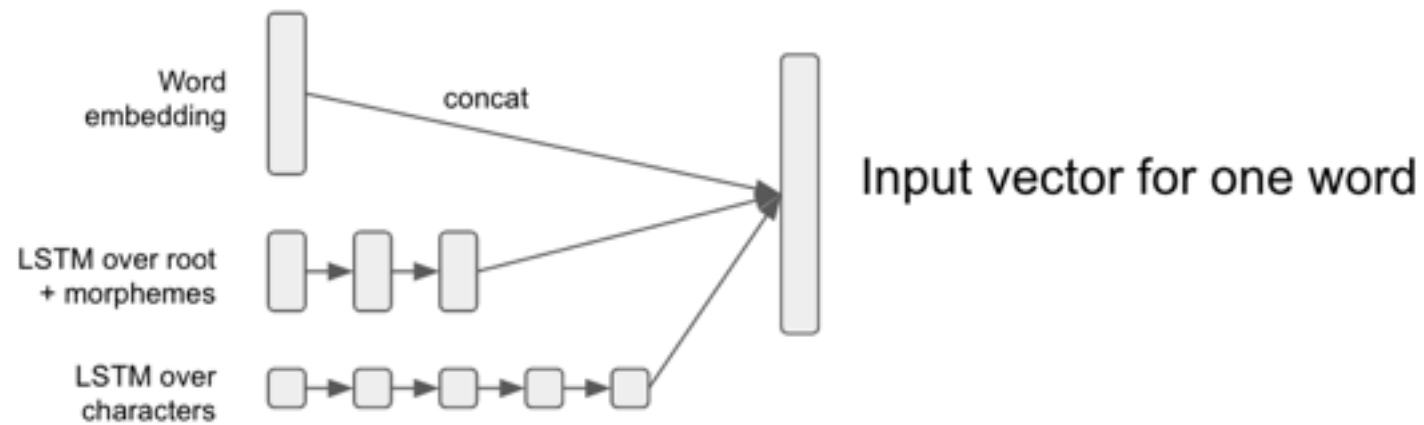
Input word representation:



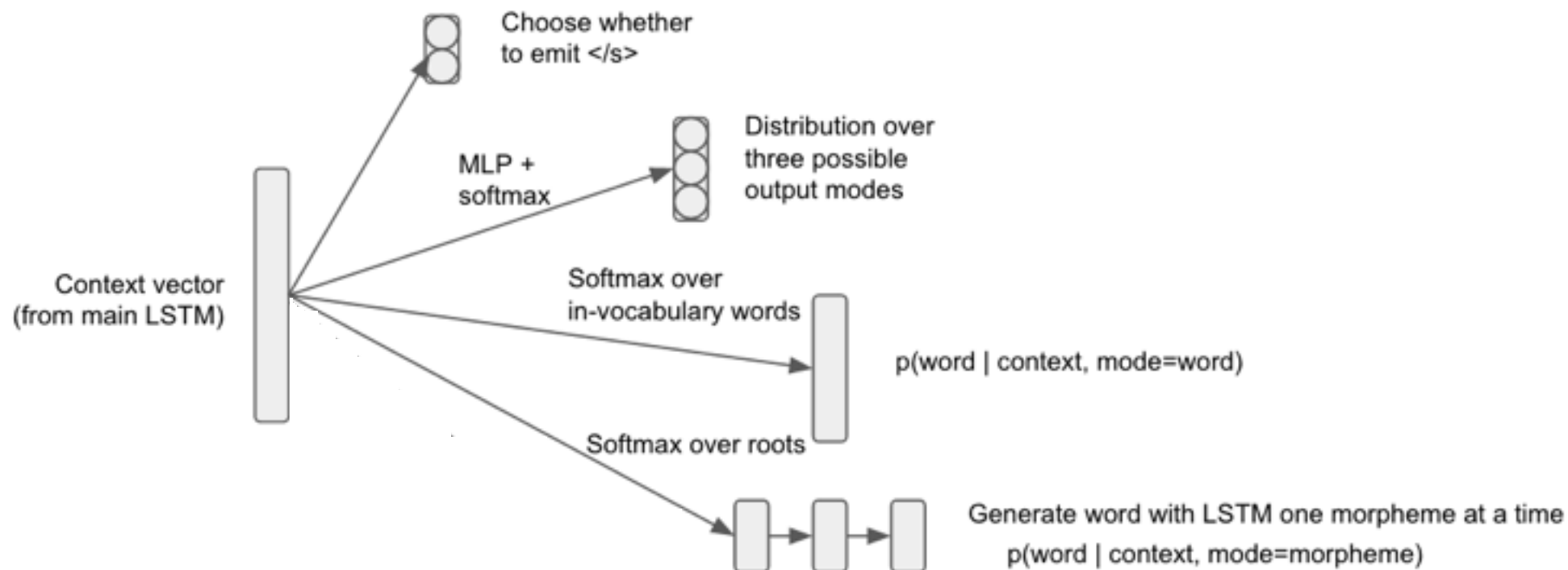
Output generation process: (marginalized)



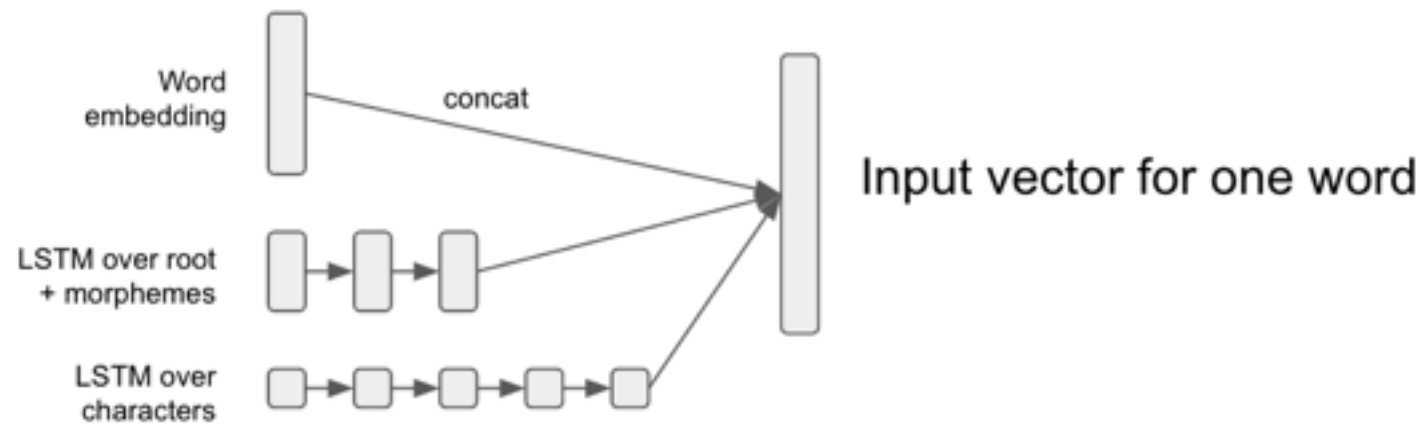
Input word representation:



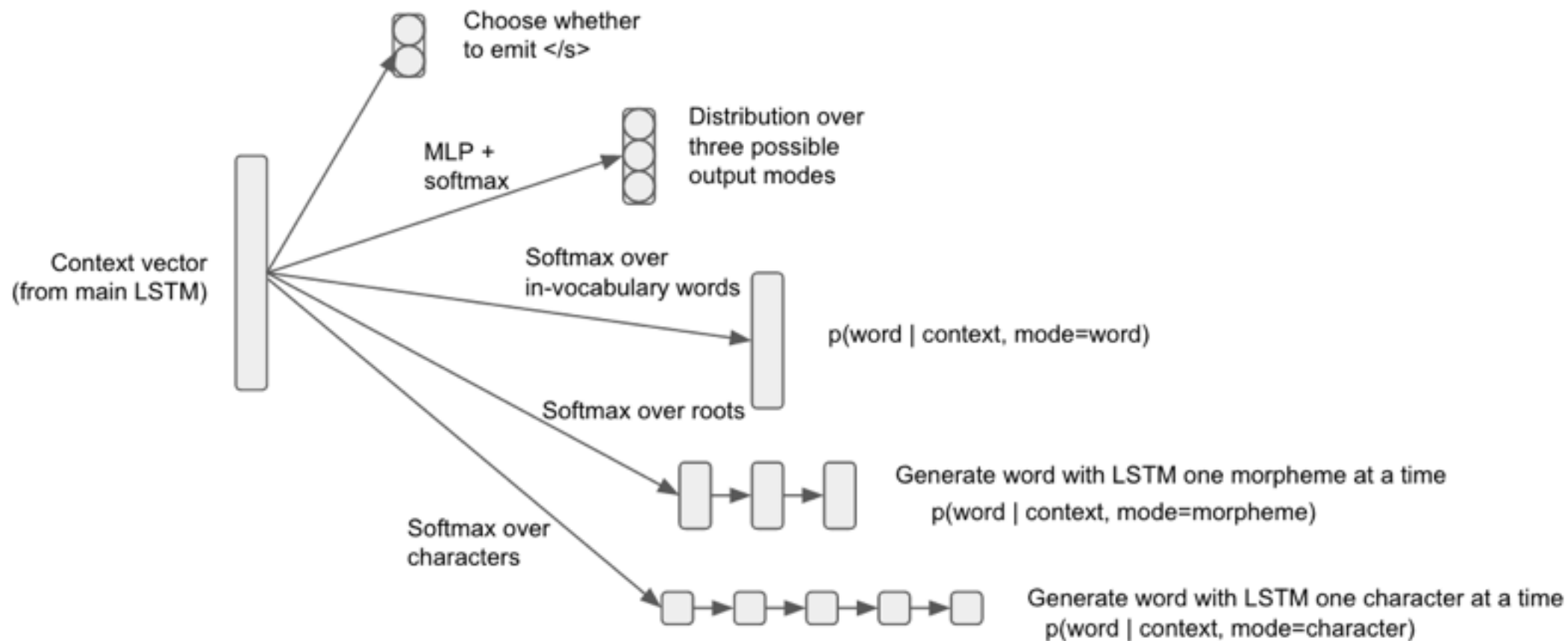
Output generation process: (marginalized)



Input word representation:



Output generation process: (marginalized)



Open Vocabulary LM

	perplexity per word
Characters	18600
Characters +Morphs	8165
Characters +Words	5021
Characters +Words +Morphs	4116

Character vs. word modeling

Summary

- Model performance is essentially equivalent in morphologically simple languages (e.g., Chinese, English)
- In morphologically rich languages (e.g., Hungarian, Turkish, Finnish), performance improvements are most pronounced
- We need **far fewer parameters** to represent words as “compositions” of characters
- Word and morpheme level information adds additional value
- **Where else could we add linguistic structural knowledge?**

Modeling syntax

Language is hierarchical

(1) a. The talk I gave did **not** appeal to **anybody**.

Modeling syntax

Language is hierarchical

- (1) a. The talk I gave did **not** appeal to **anybody**.
b. *The talk I gave appealed to **anybody**.

Modeling syntax

Language is hierarchical

- (1) a. The talk I gave did **not** appeal to **anybody**.
b. *The talk I gave appealed to **anybody**.
- 

Modeling syntax

Language is hierarchical

- (1) a. The talk I gave did **not** appeal to **anybody**.
b. *The talk I gave appealed to **anybody**.
- 
- A red bracket labeled 'NPI' connects the word 'not' to the word 'anybody' in sentence (1) a.

Generalization hypothesis: **not** must come before **anybody**

Modeling syntax

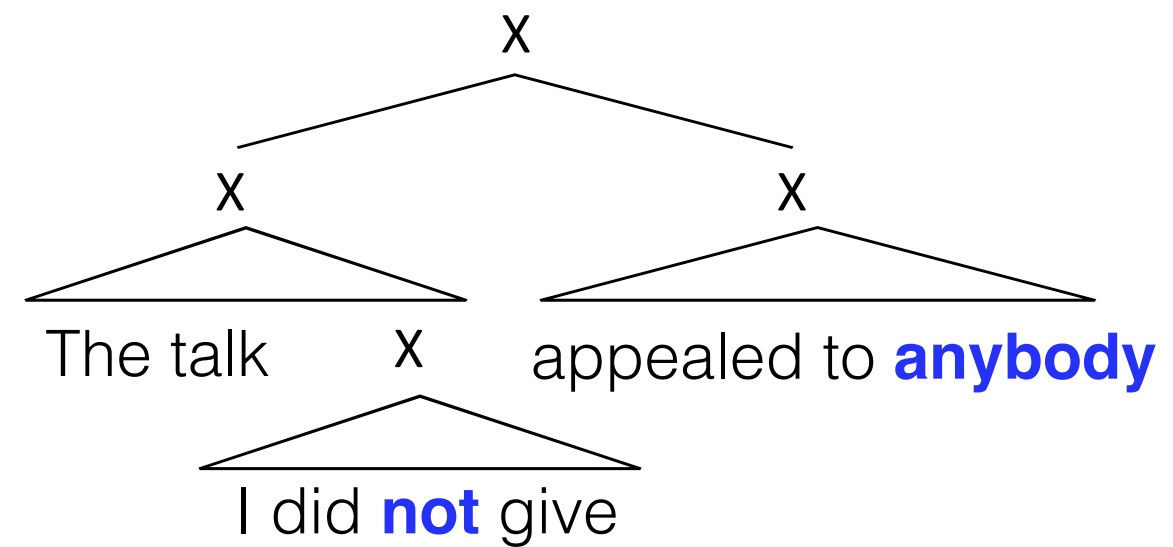
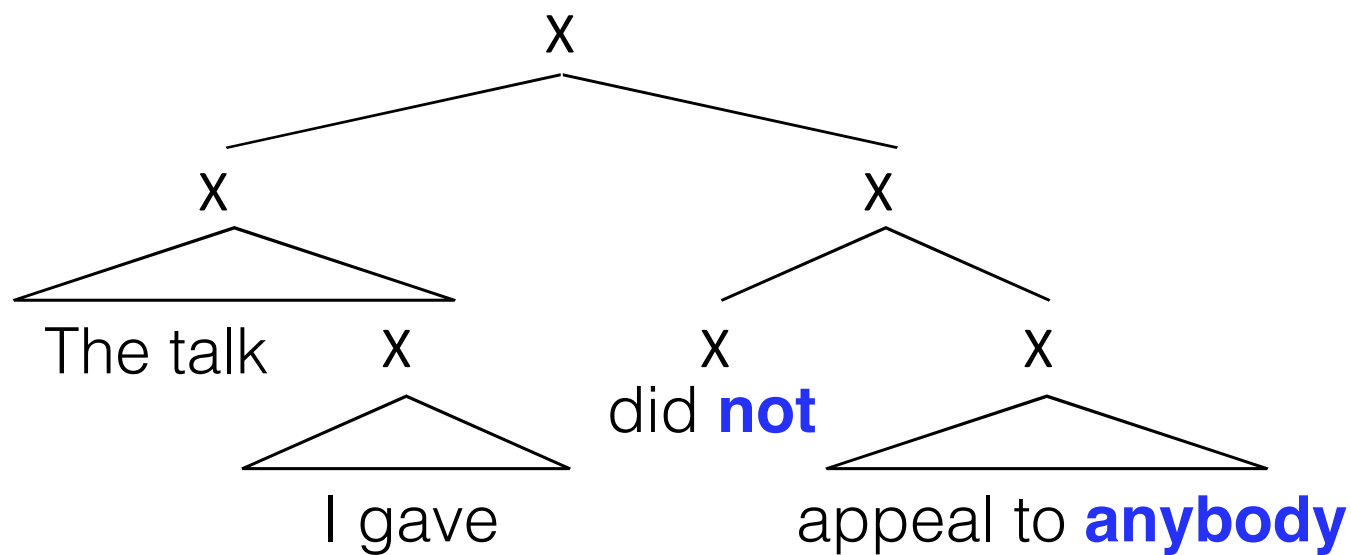
Language is hierarchical

- (1) a. The talk I gave did **not** appeal to **anybody**.
b. *The talk I gave appealed to **anybody**.
- 

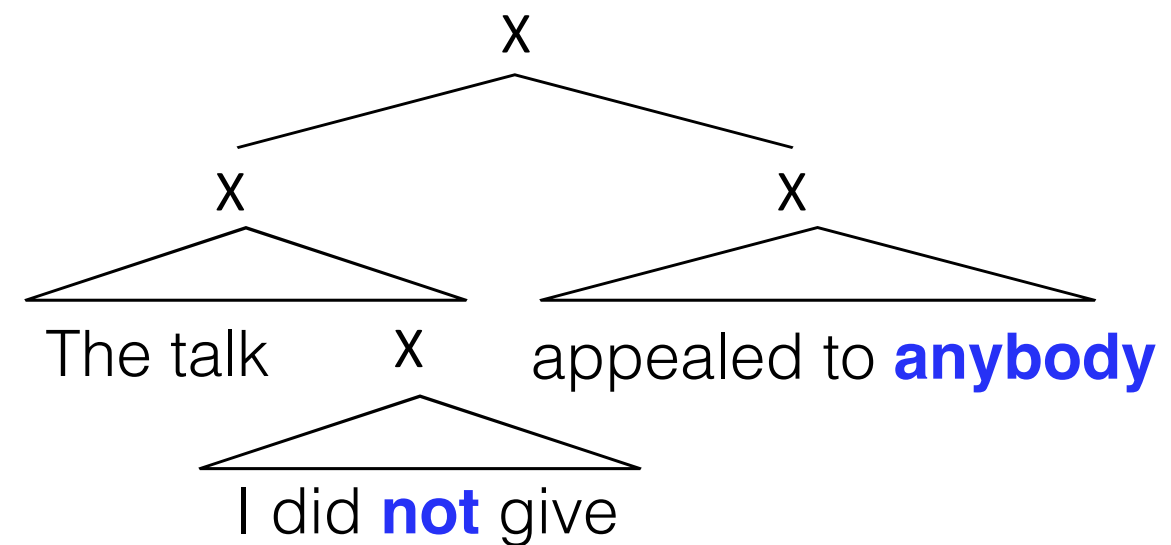
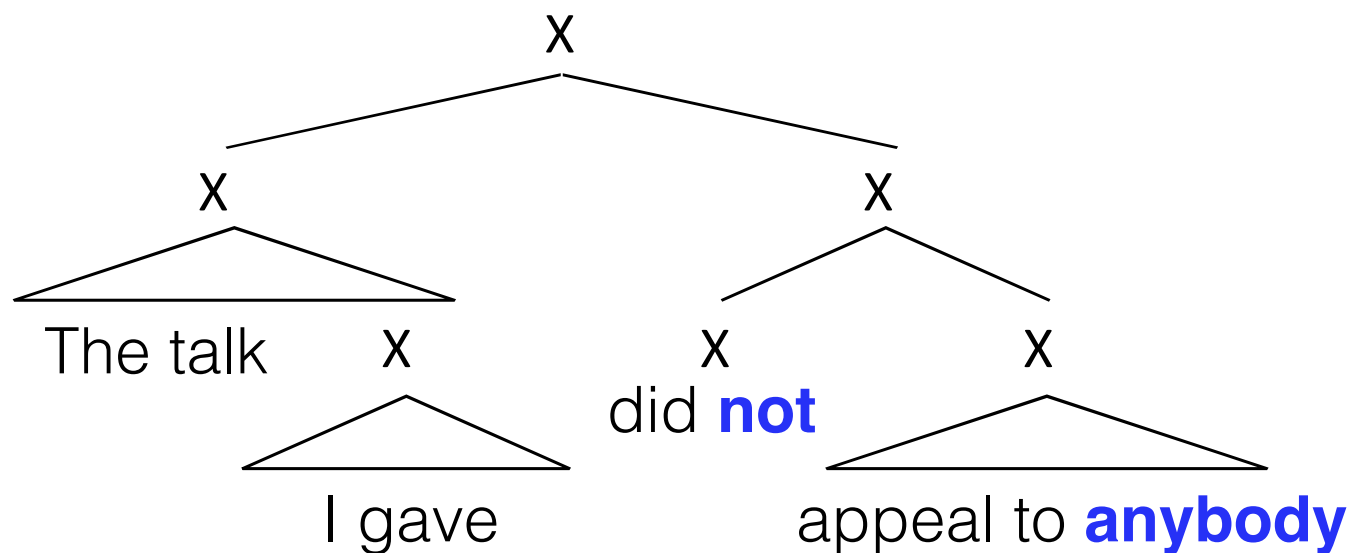
Generalization hypothesis: **not** must come before **anybody**

- (2) *The talk I did **not** give appealed to **anybody**.

Language is hierarchical

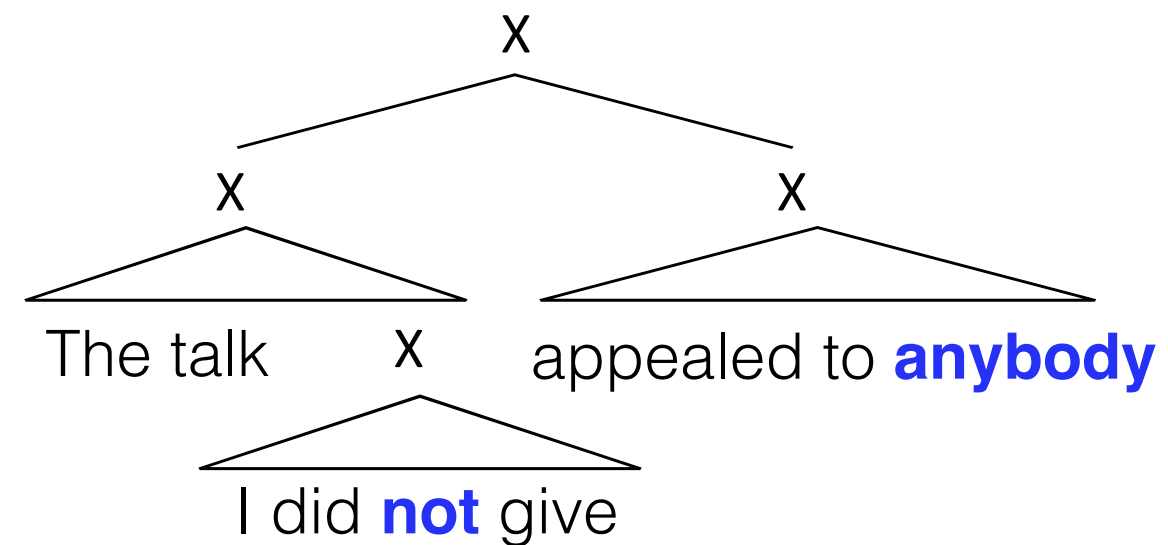
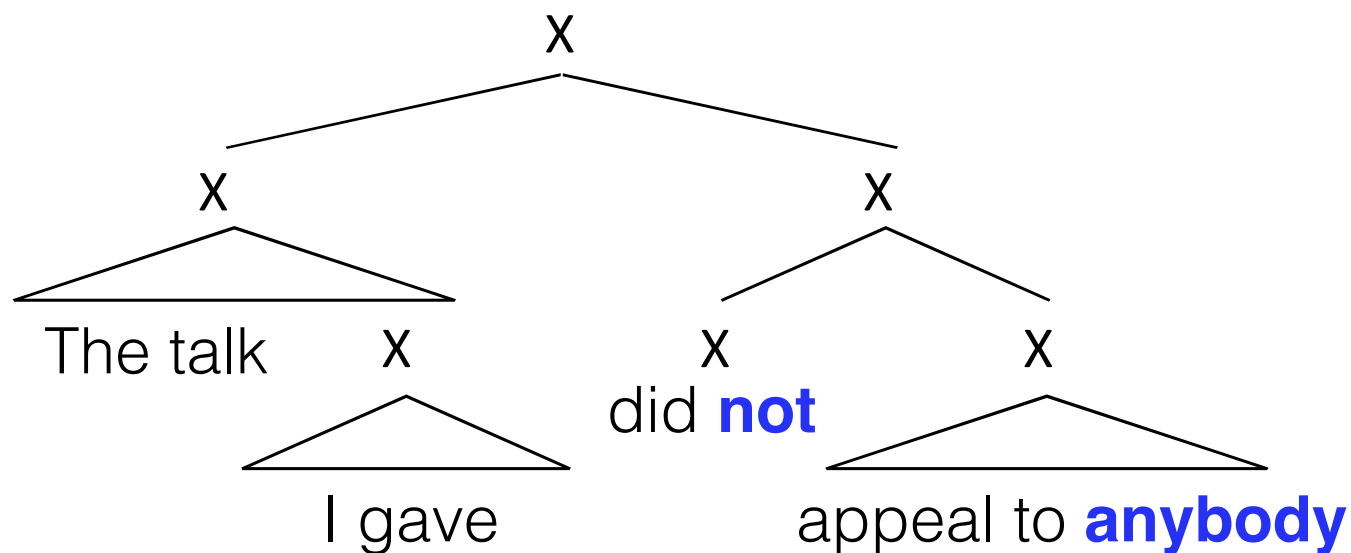


Language is hierarchical



Generalization: **not** must “structurally precede” **anybody**

Language is hierarchical



Generalization: **not** must “structurally precede” **anybody**

- many theories of the details of structure
- the psychological reality of structural sensitivity is **not** empirically controversial
- much more than NPIs follow such constraints

One theory of hierarchy

- Generate symbols sequentially using an RNN

One theory of hierarchy

- Generate symbols sequentially using an RNN
- Add some “control symbols” to rewrite the history periodically
 - Periodically “compress” a sequence into a single “constituent”
 - Augment RNN with an operation to compress recent history into a single vector (-> “reduce”)
 - RNN predicts next symbol based on the history of compressed elements and non-compressed terminals (“shift” or “generate”)
 - RNN must also predict “control symbols” that decide how big constituents are

One theory of hierarchy

- Generate symbols sequentially using an RNN
- Add some “control symbols” to rewrite the history periodically
 - Periodically “compress” a sequence into a single “constituent”
 - Augment RNN with an operation to compress recent history into a single vector (-> “reduce”)
 - RNN predicts next symbol based on the history of compressed elements and non-compressed terminals (“shift” or “generate”)
 - RNN must also predict “control symbols” that decide how big constituents are
- We call such models **recurrent neural network grammars**.

Terminals

Stack

Action

Terminals

Stack

Action

NT(S)

Terminals

Stack

Action

NT(S)

(S

Terminals

Stack

Action

NT(S)

(S

NT(NP)

Terminals

Stack

Action

NT(S)

NT(NP)

(S

(S

(NP

Terminals

Stack

Action

NT(S)

NT(NP)

GEN(*The*)

(S

(S

(NP

Terminals

Stack

Action

The

(S (NP *The*

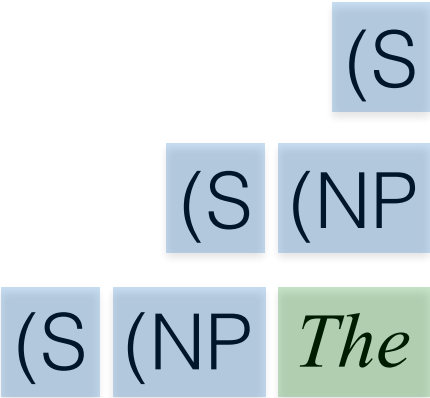
NT(S)
NT(NP)
GEN(*The*)

Terminals

Stack

Action

The



NT(S)

NT(NP)

GEN(The)

GEN(hungry)

Terminals

Stack

Action

NT(S)

NT(NP)

GEN(*The*)

GEN(*hungry*)

The

The hungry

(S

(S (NP

(S (NP *The*

(S (NP *The hungry*

Terminals

Stack

Action

NT(S)

NT(NP)

GEN(*The*)

GEN(*hungry*)

GEN(*cat*)

The

The hungry

(S

(S (NP

(S (NP *The*

(S (NP *The hungry*

Terminals

Stack

Action

NT(S)

NT(NP)

GEN(*The*)

GEN(*hungry*)

GEN(*cat*)

The

The hungry

The hungry cat

(S

(S (NP

(S (NP *The*

(S (NP *The hungry*

(S (NP *The hungry cat*

Terminals

Stack

Action

NT(S)

NT(NP)

GEN(*The*)

GEN(*hungry*)

GEN(*cat*)

REDUCE

The

The hungry

The hungry cat

(S

(S (NP

(S (NP *The*

(S (NP *The hungry*

(S (NP *The hungry cat*

Terminals

Stack

Action

NT(S)

NT(NP)

GEN(*The*)

GEN(*hungry*)

GEN(*cat*)

REDUCE

The

The hungry

The hungry cat

The hungry cat

(S

(S (NP

(S (NP *The*

(S (NP *The hungry*

(S (NP *The hungry cat*

(S (NP *The hungry cat*)

Terminals	Stack	Action
		NT (S)
	(S	NT (NP)
	(S (NP	GEN (<i>The</i>)
<i>The</i>	(S (NP <i>The</i>	GEN (<i>hungry</i>)
<i>The hungry</i>	(S (NP <i>The hungry</i>	GEN (<i>cat</i>)
<i>The hungry cat</i>	(S (NP <i>The hungry cat</i>	REDUCE
<i>The hungry cat</i>	(S (NP <i>The hungry cat</i>)	
	(S (NP <i>The hungry cat</i>)	
Compress “The hungry cat” into a single composite symbol		

Terminals

Stack

Action

NT(S)

NT(NP)

GEN(*The*)

GEN(*hungry*)

GEN(*cat*)

REDUCE

The

The hungry

The hungry cat

The hungry cat

(S

(S (NP

(S (NP *The*

(S (NP *The hungry*

(S (NP *The hungry cat*

(S (NP *The hungry cat*)

Terminals

Stack

Action

NT(S)

NT(NP)

GEN(*The*)

GEN(*hungry*)

GEN(*cat*)

REDUCE

NT(VP)

The

The hungry

The hungry cat

The hungry cat

(S

(S (NP

(S (NP *The*

(S (NP *The hungry*

(S (NP *The hungry cat*

(S (NP *The hungry cat*)

Terminals

Stack

Action

NT(S)

NT(NP)

GEN(*The*)

GEN(*hungry*)

GEN(*cat*)

REDUCE

NT(VP)

The

The hungry

The hungry cat

The hungry cat

The hungry cat

(S

(S (NP

(S (NP *The*

(S (NP *The hungry*

(S (NP *The hungry cat*

(S (NP *The hungry cat*)

(S (NP *The hungry cat*) (VP

Terminals

Stack

Action

NT(S)

NT(NP)

GEN(*The*)

GEN(*hungry*)

GEN(*cat*)

REDUCE

NT(VP)

GEN(*meows*)

The

The hungry

The hungry cat

The hungry cat

The hungry cat

(S

(S (NP

(S (NP *The*

(S (NP *The hungry*

(S (NP *The hungry cat*

(S (NP *The hungry cat*)

(S (NP *The hungry cat*) (VP

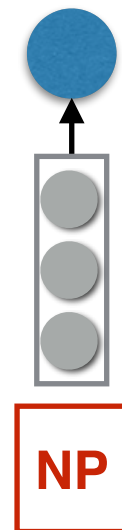
Terminals	Stack	Action
		NT(S)
	(S	NT(NP)
	(S (NP	GEN(The)
The	(S (NP The	GEN(hungry)
The hungry	(S (NP The hungry	GEN(cat)
The hungry cat	(S (NP The hungry cat	REDUCE
The hungry cat	(S (NP The hungry cat)	NT(VP)
The hungry cat	(S (NP The hungry cat) (VP	GEN(meows)
The hungry cat meows	(S (NP The hungry cat) (VP meows	REDUCE
The hungry cat meows	(S (NP The hungry cat) (VP meows)	GEN(.)
The hungry cat meows .	(S (NP The hungry cat) (VP meows) .	REDUCE
The hungry cat meows .	(S (NP The hungry cat) (VP meows) .)	

Syntactic Composition

Need representation for: (NP *The hungry cat*)

Syntactic Composition

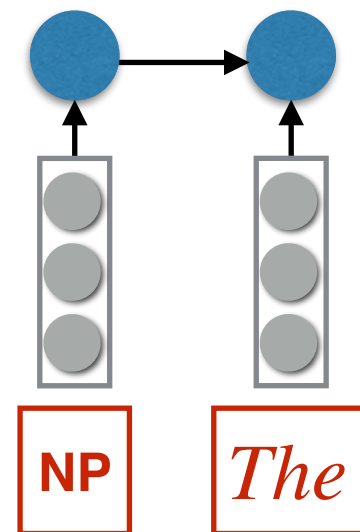
Need representation for: (NP *The hungry cat*)



What head type? 

Syntactic Composition

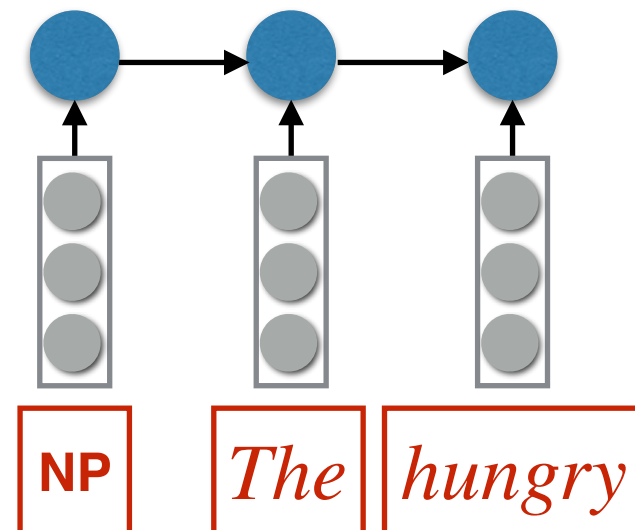
Need representation for: (NP *The hungry cat*)



What head type? ↗

Syntactic Composition

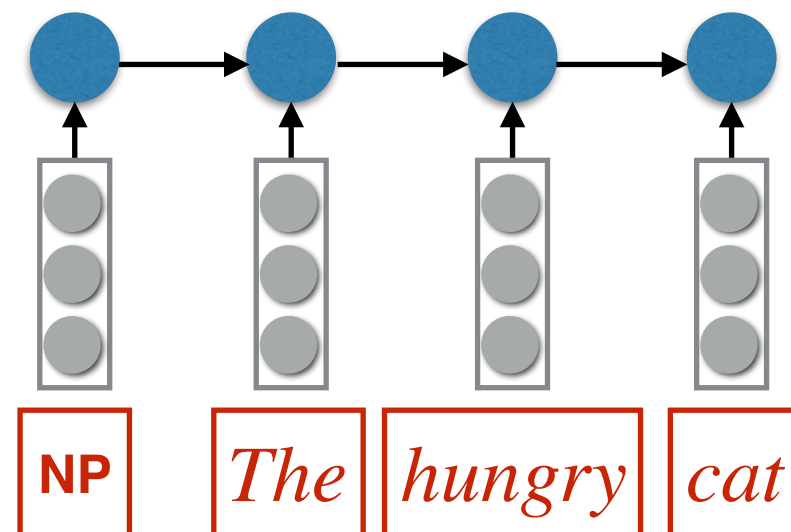
Need representation for: (NP *The hungry cat*)



What head type? ↗

Syntactic Composition

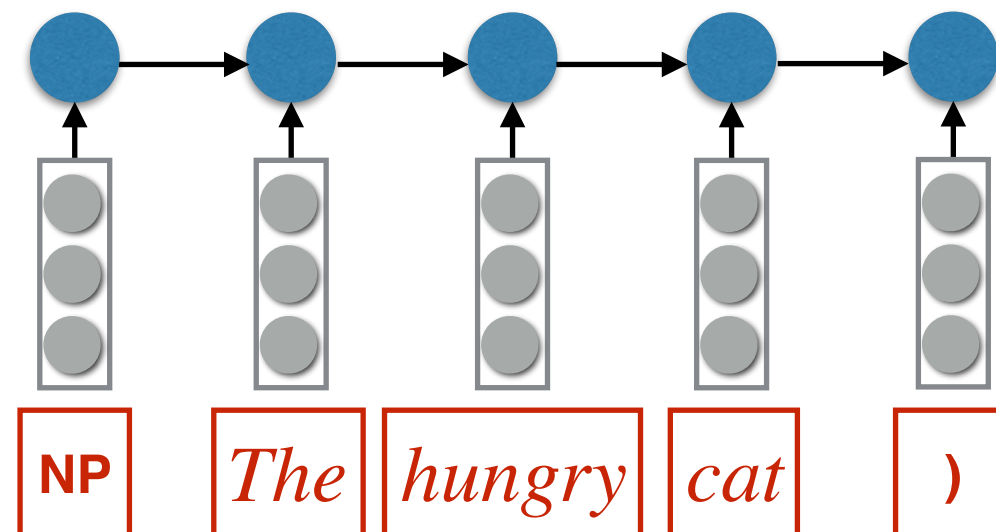
Need representation for: (NP *The hungry cat*)



What head type? ↗

Syntactic Composition

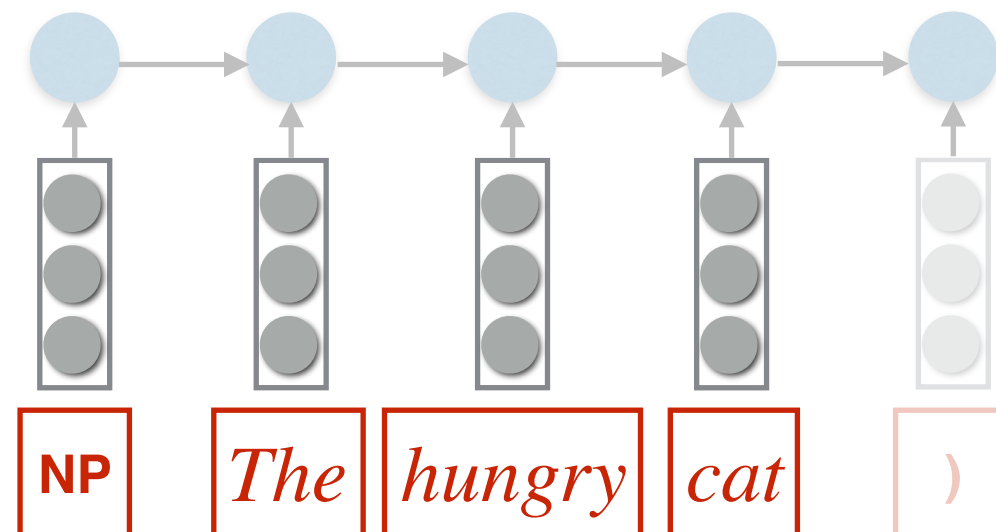
Need representation for: (NP *The hungry cat*)



What head type? ↗

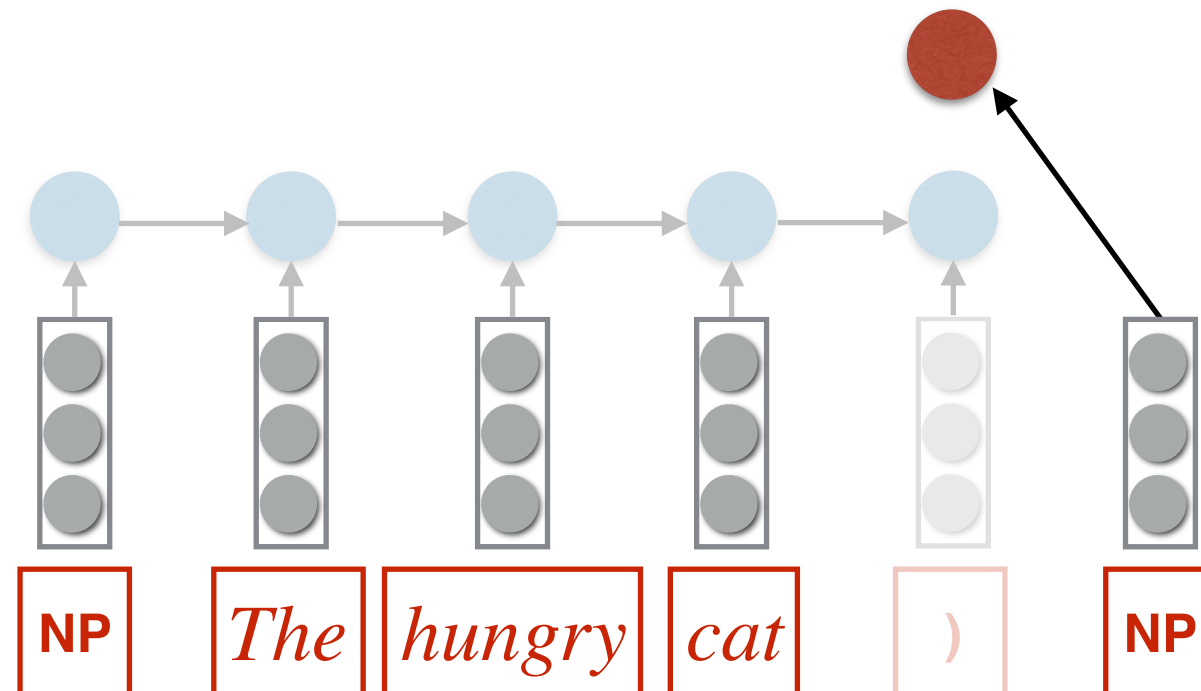
Syntactic Composition

Need representation for: (NP *The hungry cat*)



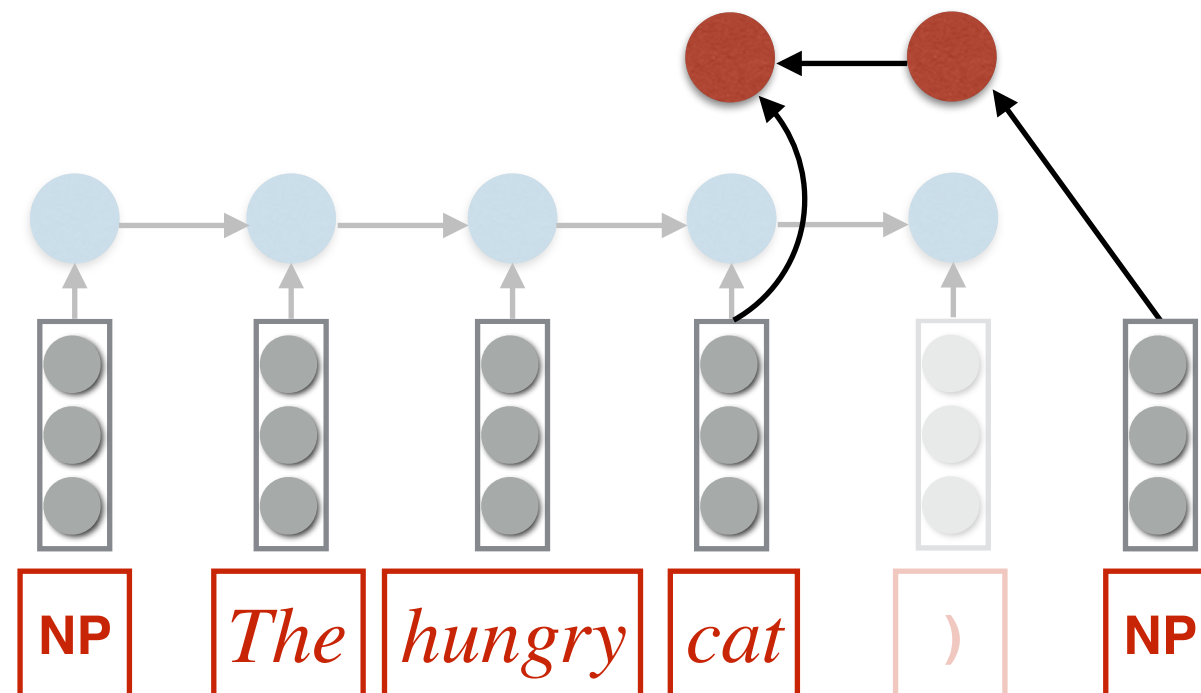
Syntactic Composition

Need representation for: (NP *The hungry cat*)



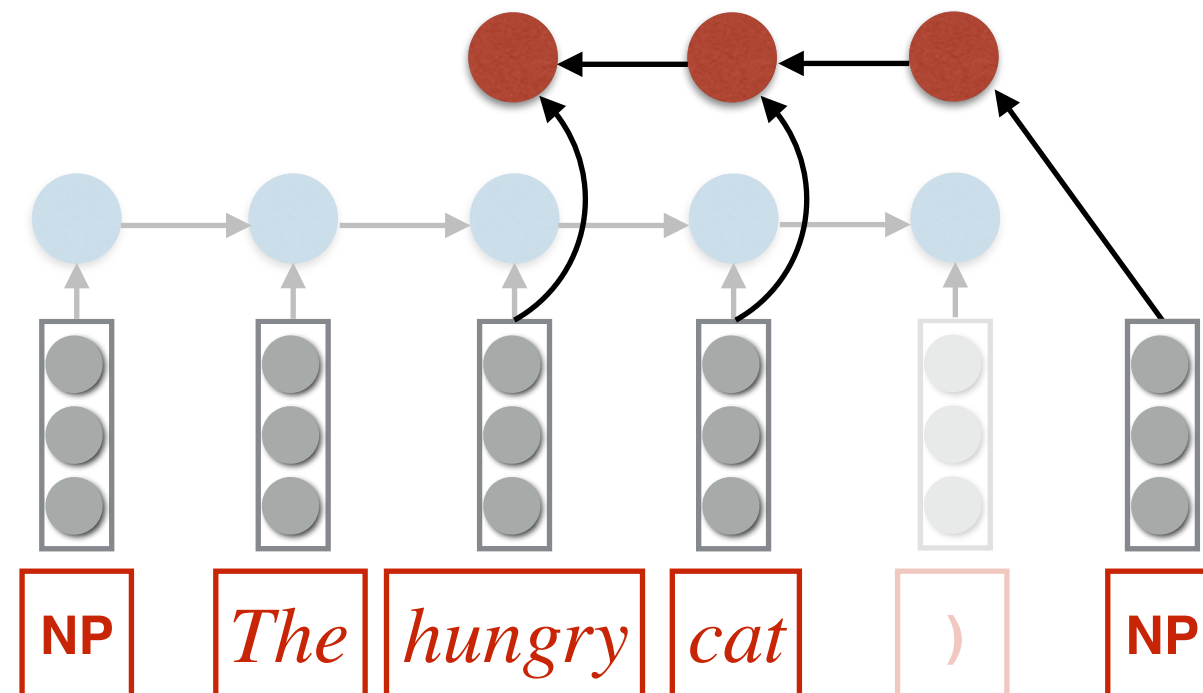
Syntactic Composition

Need representation for: (NP *The hungry cat*)



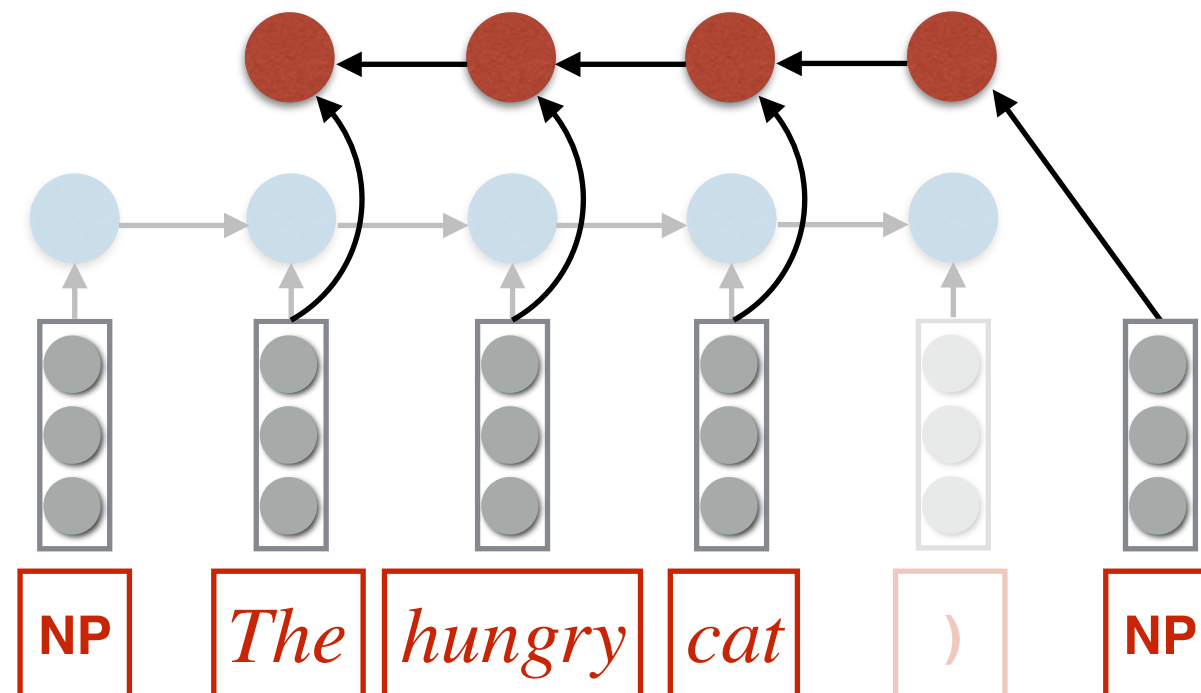
Syntactic Composition

Need representation for: (NP *The hungry cat*)



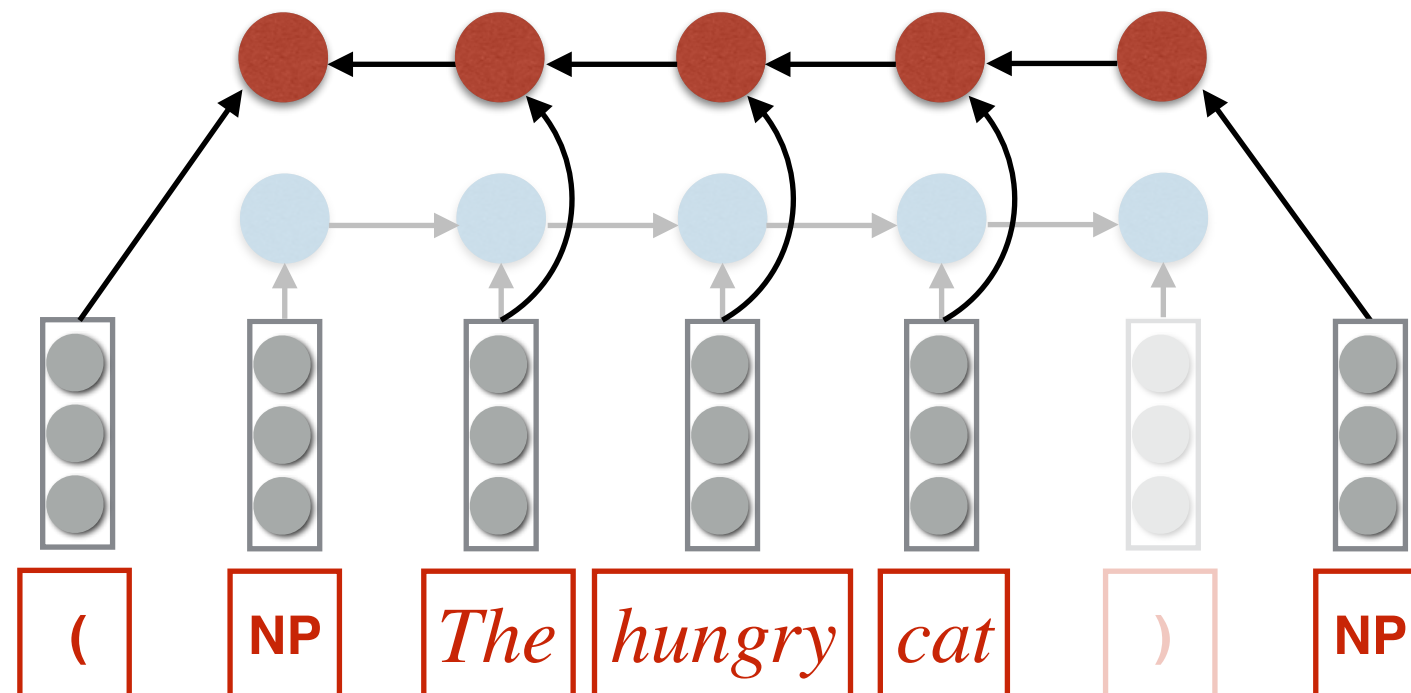
Syntactic Composition

Need representation for: (NP *The hungry cat*)



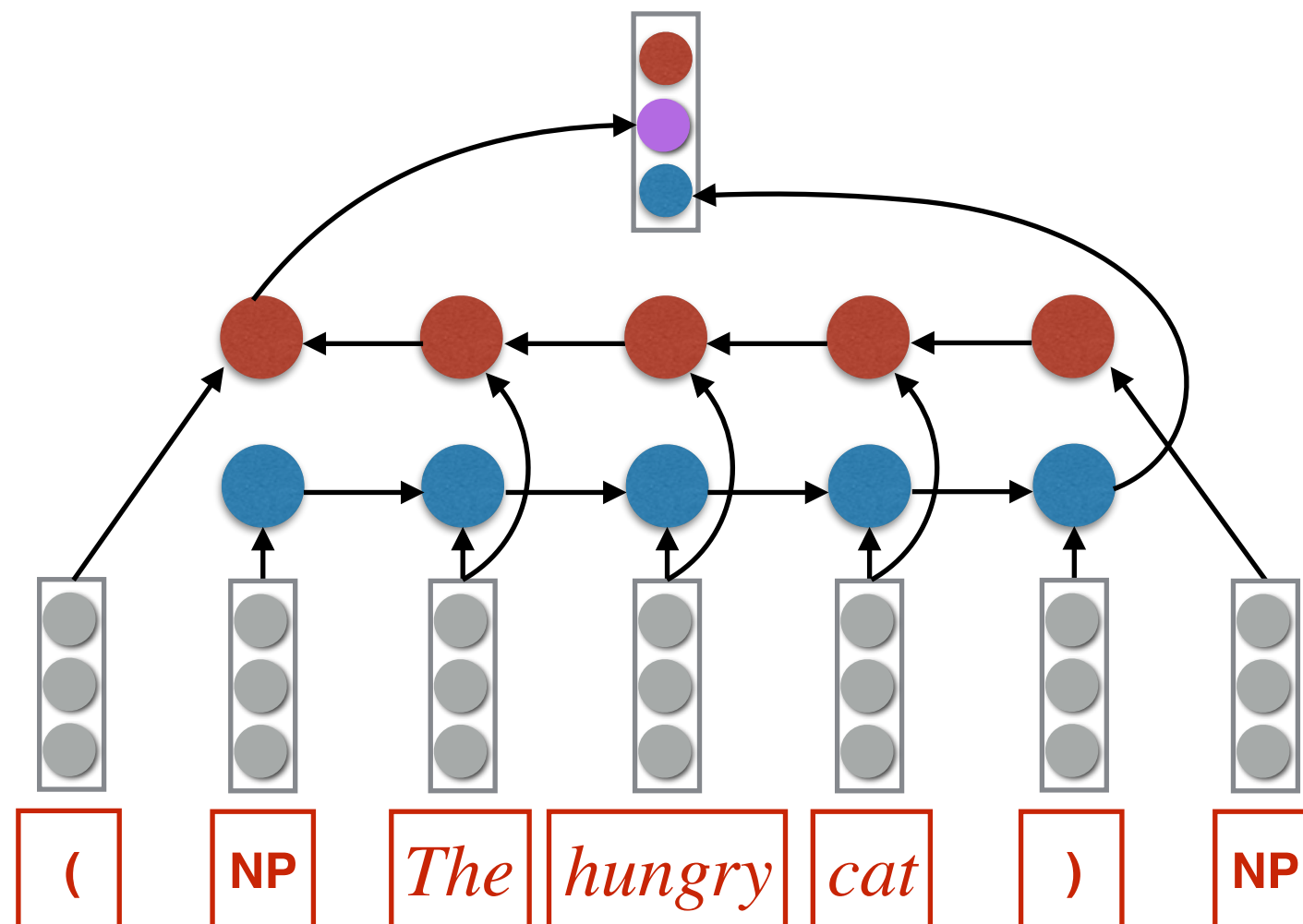
Syntactic Composition

Need representation for: (NP *The hungry cat*)



Syntactic Composition

Need representation for: (NP *The hungry cat*)

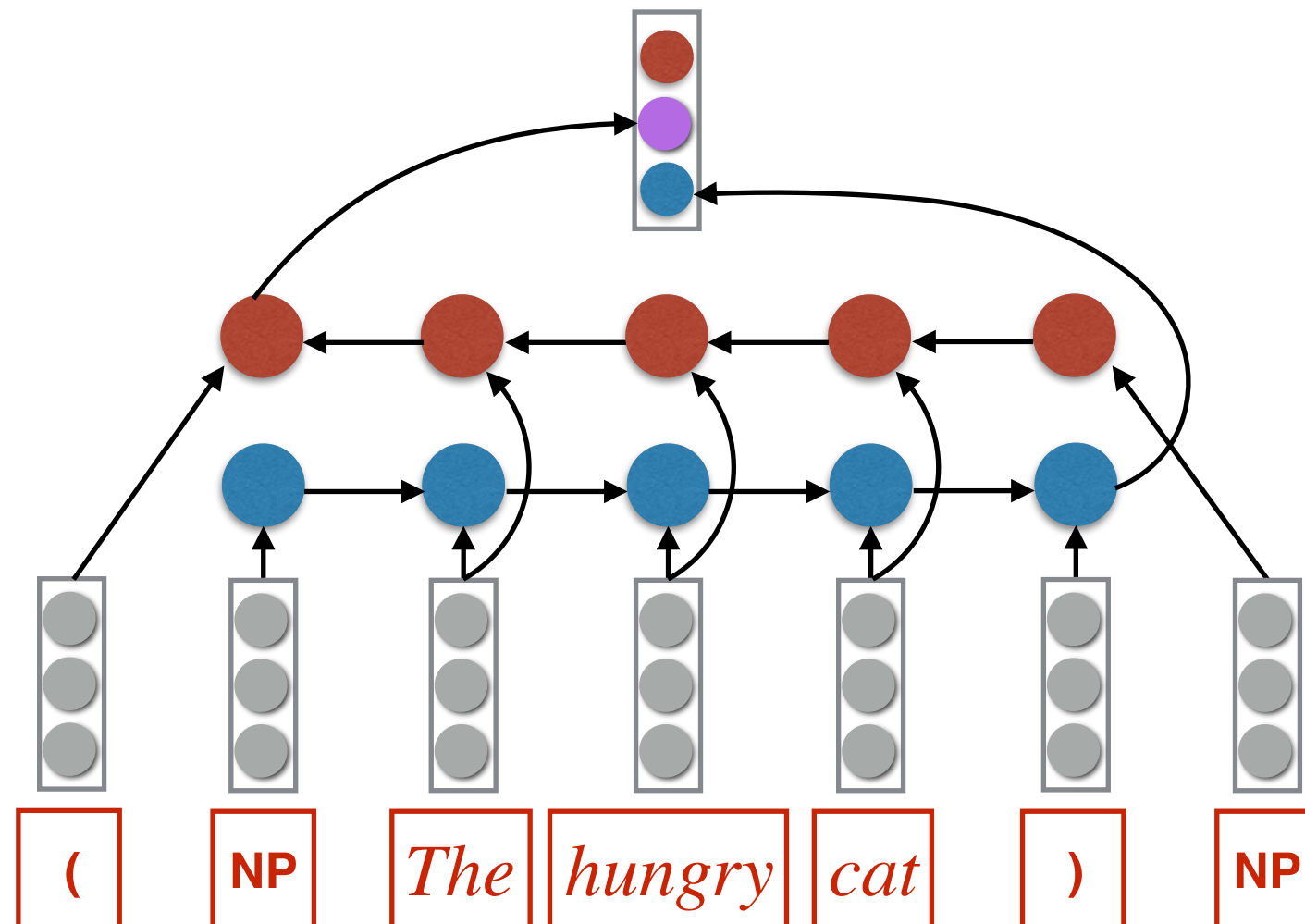


Recursion

Need representation for:

(NP *The hungry cat*)

(NP *The (ADJP very hungry) cat*)

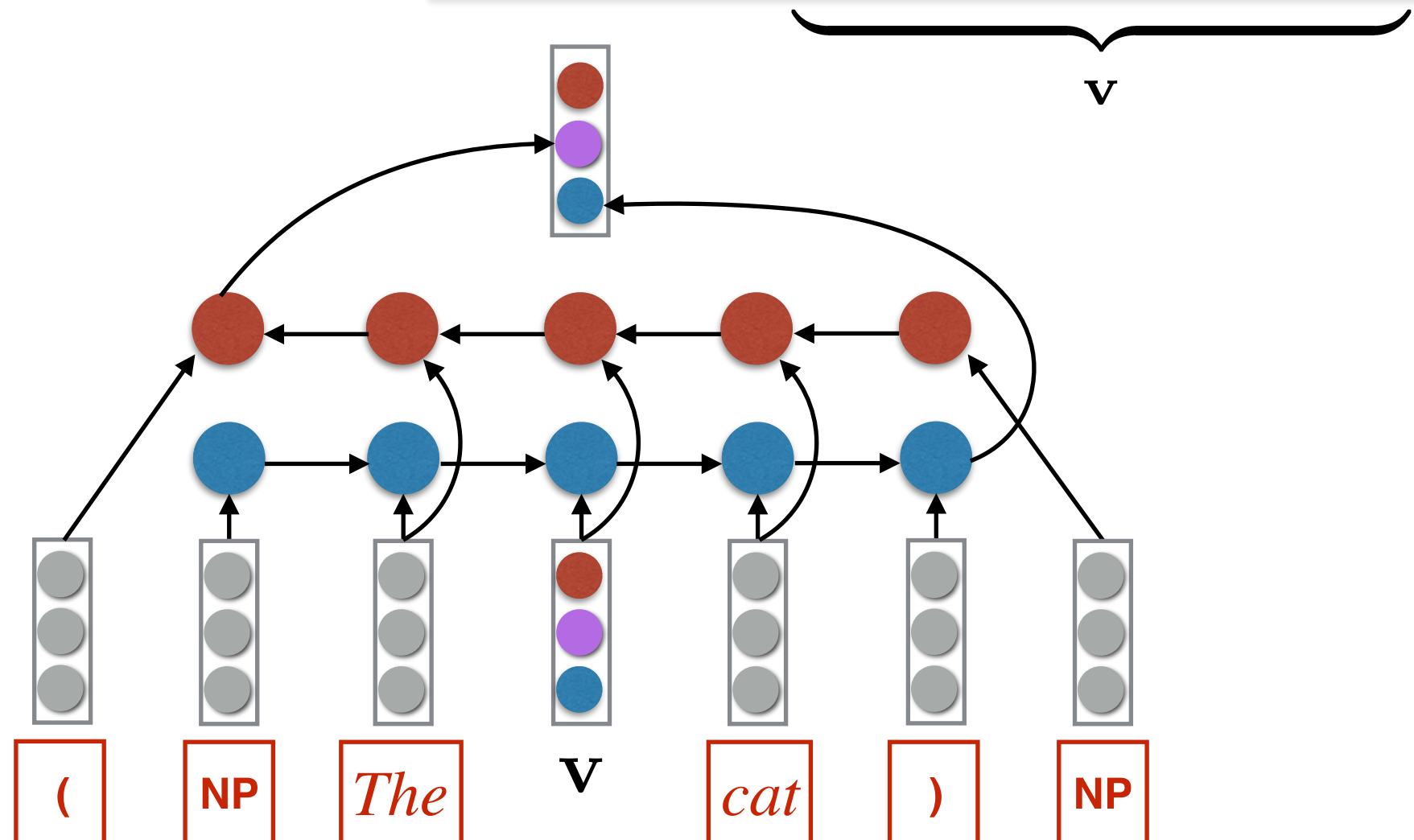


Recursion

Need representation for:

(NP *The hungry cat*)

(NP *The (ADJP very hungry) cat*)



Syntactic Composition

- Inspired by Socher et al (2011, 2012 ...)
 - words and constituents embedded in same space
- Composition functions designed to
 - capture linguistic notion of **headedness**
(LSTMs know what type of head they are looking for while they traverse children)
 - support any number of children
 - are learned via backpropagation through structure

Implementing RNNs

Parameter Estimation

- RNNs jointly model sequences of words together with a “tree structure”, $p_{\theta}(\mathbf{x}, \mathbf{y})$
- Any parse tree can be converted to a sequence of actions (depth first traversal) and vice versa (subject to wellformedness constraints)
 - We use trees from the Penn Treebank
- We could treat the non-generation actions as latent variables or learn them with RL, effectively making this a problem of *grammar induction*. Future work...

Implementing RNNGs

Inference

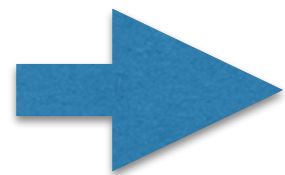
- An RNNG is a joint distribution $p(\mathbf{x}, \mathbf{y})$ over strings (\mathbf{x}) and parse trees (\mathbf{y})
- We are interested in two inference questions:
 - What is $p(\mathbf{x})$ for a given \mathbf{x} ? [**language modeling**]
 - What is $\max_{\mathbf{y}} p(\mathbf{y} \mid \mathbf{x})$ for a given \mathbf{x} ? [**parsing**]
- Unfortunately, the dynamic programming algorithms we often rely on are of no help here
- We can use importance sampling to do both by sampling from a discriminatively trained model

English PTB (Parsing)

	Type	F1
Petrov and Klein (2007)	G	90.1
Shindo et al (2012) Single model	G	91.1
Shindo et al (2012) Ensemble	~G	92.4
Vinyals et al (2015) PTB only	D	90.5
Vinyals et al (2015) Ensemble	S	92.8
<i>Discriminative</i>	D	89.8
<i>Generative (IS)</i>	G	92.4

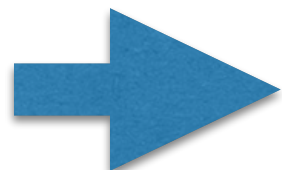
English PTB (Parsing)

	Type	F1
Petrov and Klein (2007)	G	90.1
Shindo et al (2012) Single model	G	91.1
Shindo et al (2012) Ensemble	~G	92.4
Vinyals et al (2015) PTB only	D	90.5
Vinyals et al (2015) Ensemble	S	92.8
<i>Discriminative</i>	D	89.8
<i>Generative (IS)</i>	G	92.4



English PTB (Parsing)

	Type	F1
Petrov and Klein (2007)	G	90.1
Shindo et al (2012) Single model	G	91.1
Shindo et al (2012) Ensemble	~G	92.4
Vinyals et al (2015) PTB only	D	90.5
Vinyals et al (2015) Ensemble	S	92.8
<i>Discriminative</i>	D	89.8
<i>Generative (IS)</i>	G	92.4



English PTB (LM)

	Perplexity
5-gram IKN	169.3
LSTM + Dropout	113.4
Generative (IS)	102.4

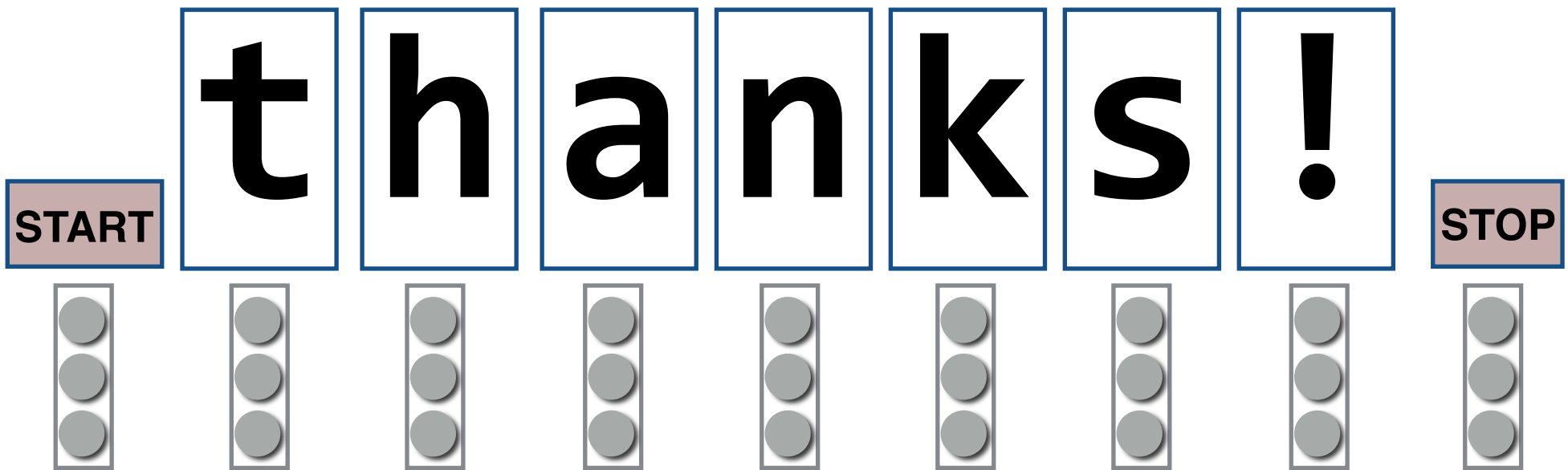
Chinese CTB (LM)

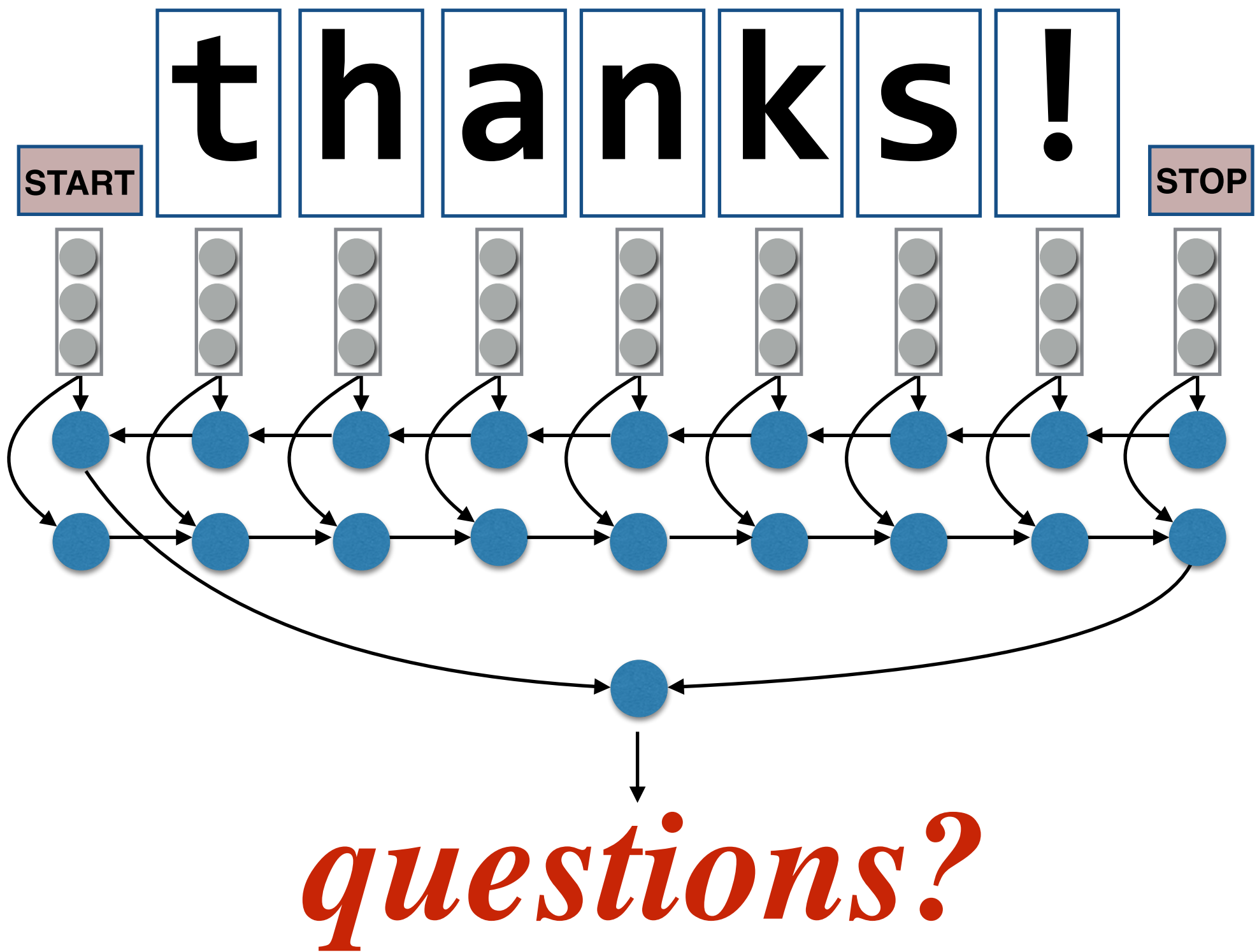
	Perplexity
5-gram IKN	255.2
LSTM + Dropout	207.3
Generative (IS)	171.9

This Talk, In a Nutshell

- **Facts about language:**
 - **Arbitrariness** and **compositionality** exist at all levels
 - Language is sensitive to **hierarchy**, not **strings**
- **My work's hypothesis:**

Models designed with these considerations structure explicit will outperform models that don't





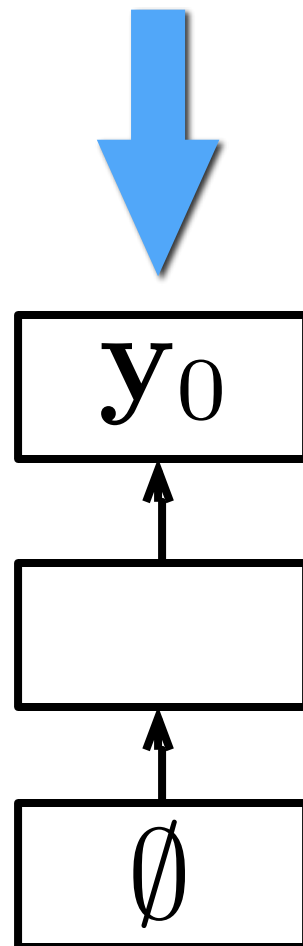
Implementing RNNs

Stack RNNs

- Augment a sequential RNN with a **stack pointer**
- Two constant-time operations
 - **push** - read input, add to top of stack, connect to current location of the stack pointer
 - **pop** - move stack pointer to its parent
- A **summary** of stack contents is obtained by accessing the output of the RNN at location of the stack pointer
- Note: **push** and **pop** are discrete actions here (*cf.* Grefenstette et al., 2015)

Implementing RNNs

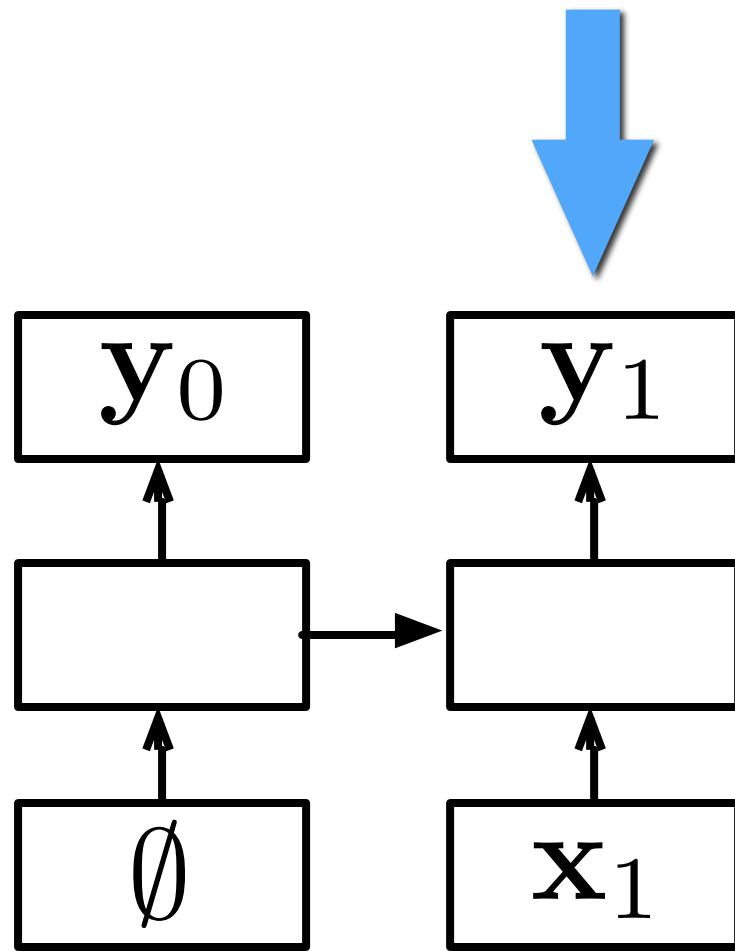
Stack RNNs



PUSH

Implementing RNNGs

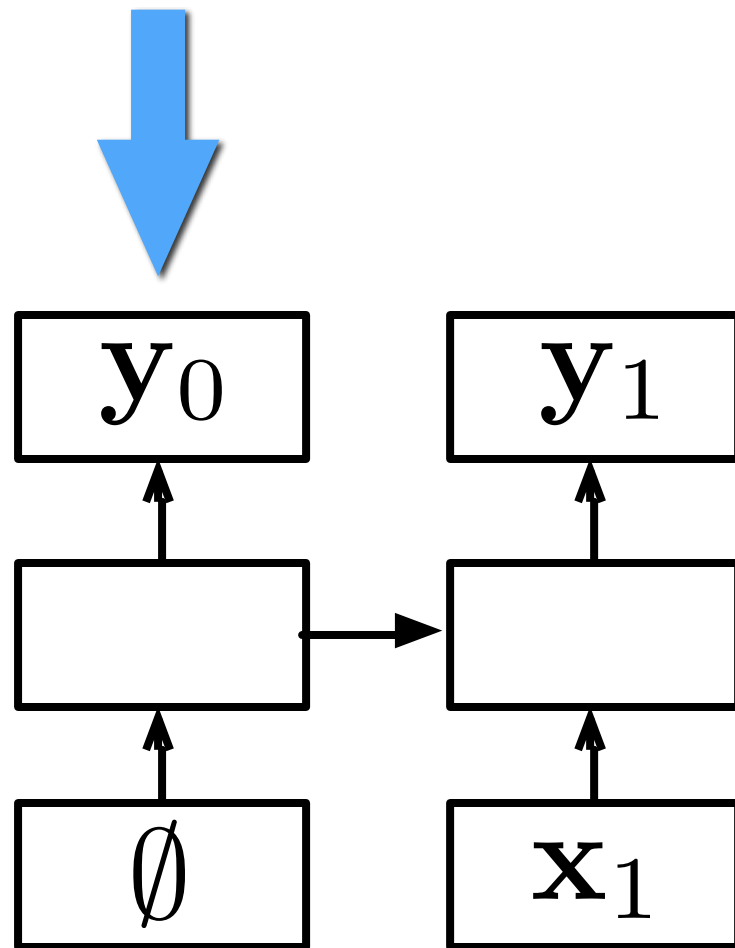
Stack RNNs



POP

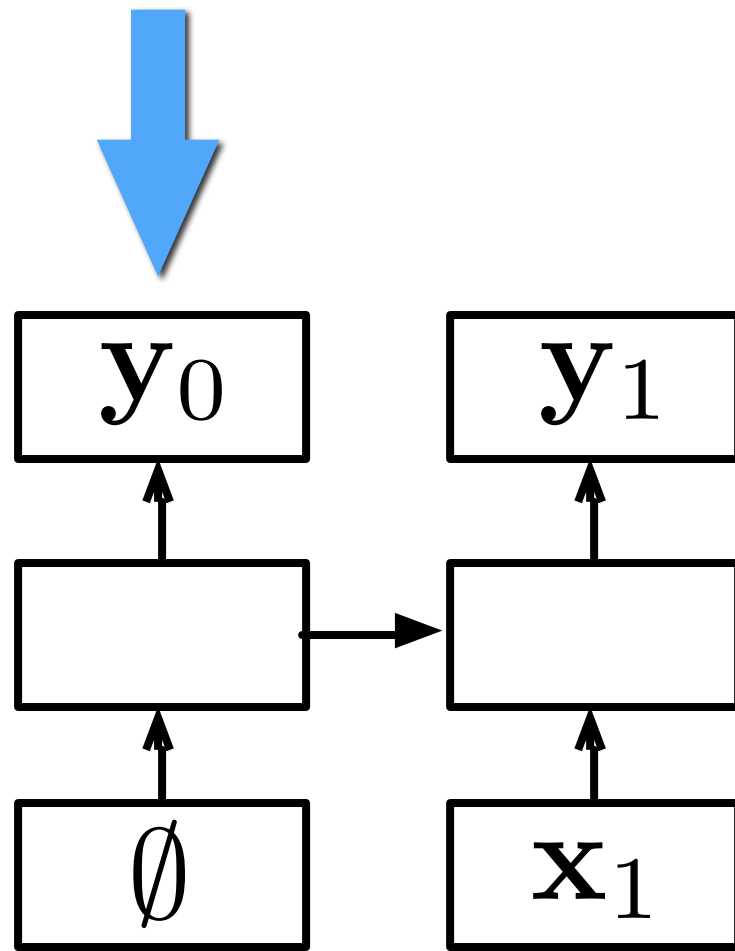
Implementing RNNs

Stack RNNs



Implementing RNNs

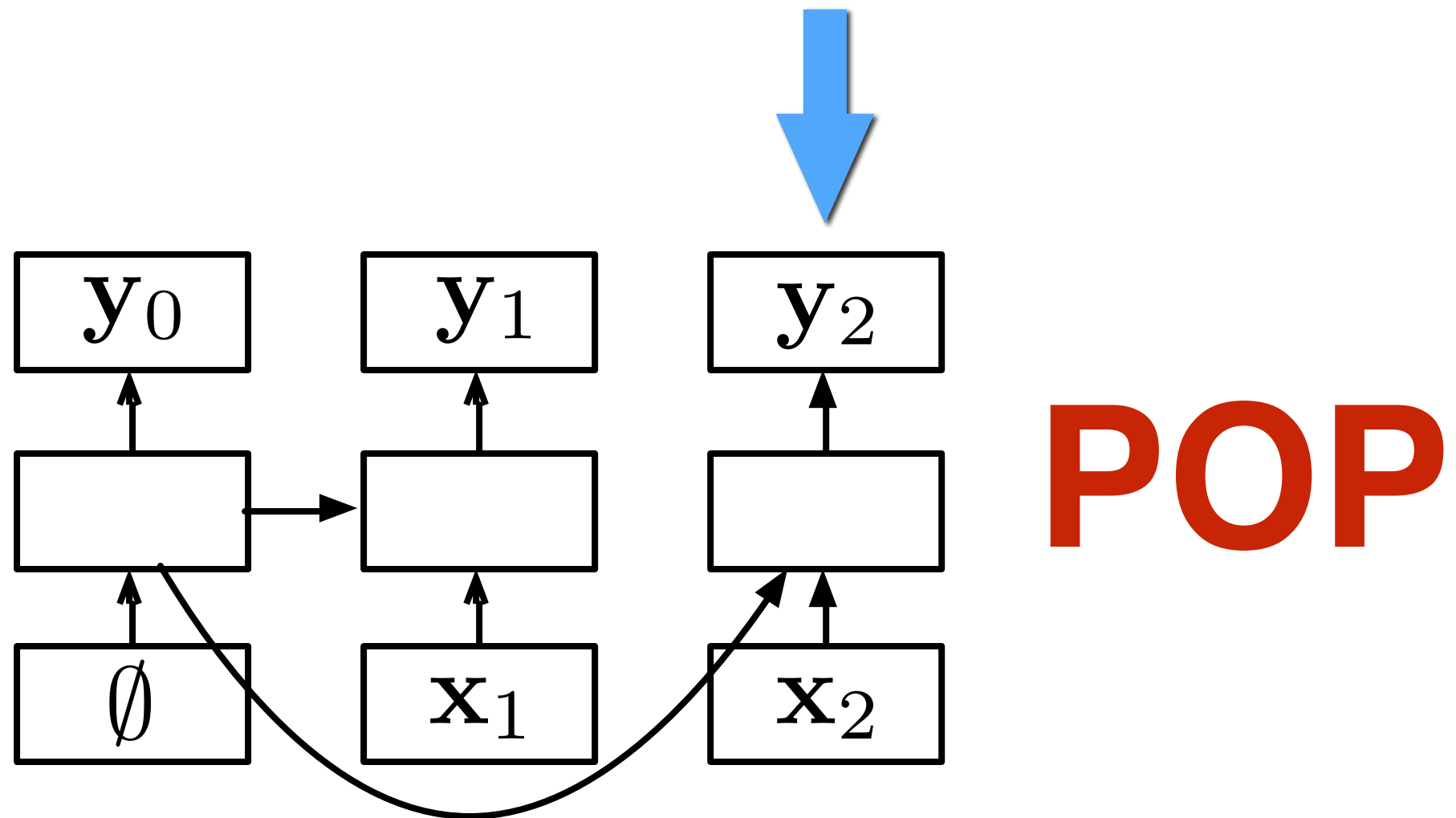
Stack RNNs



PUSH

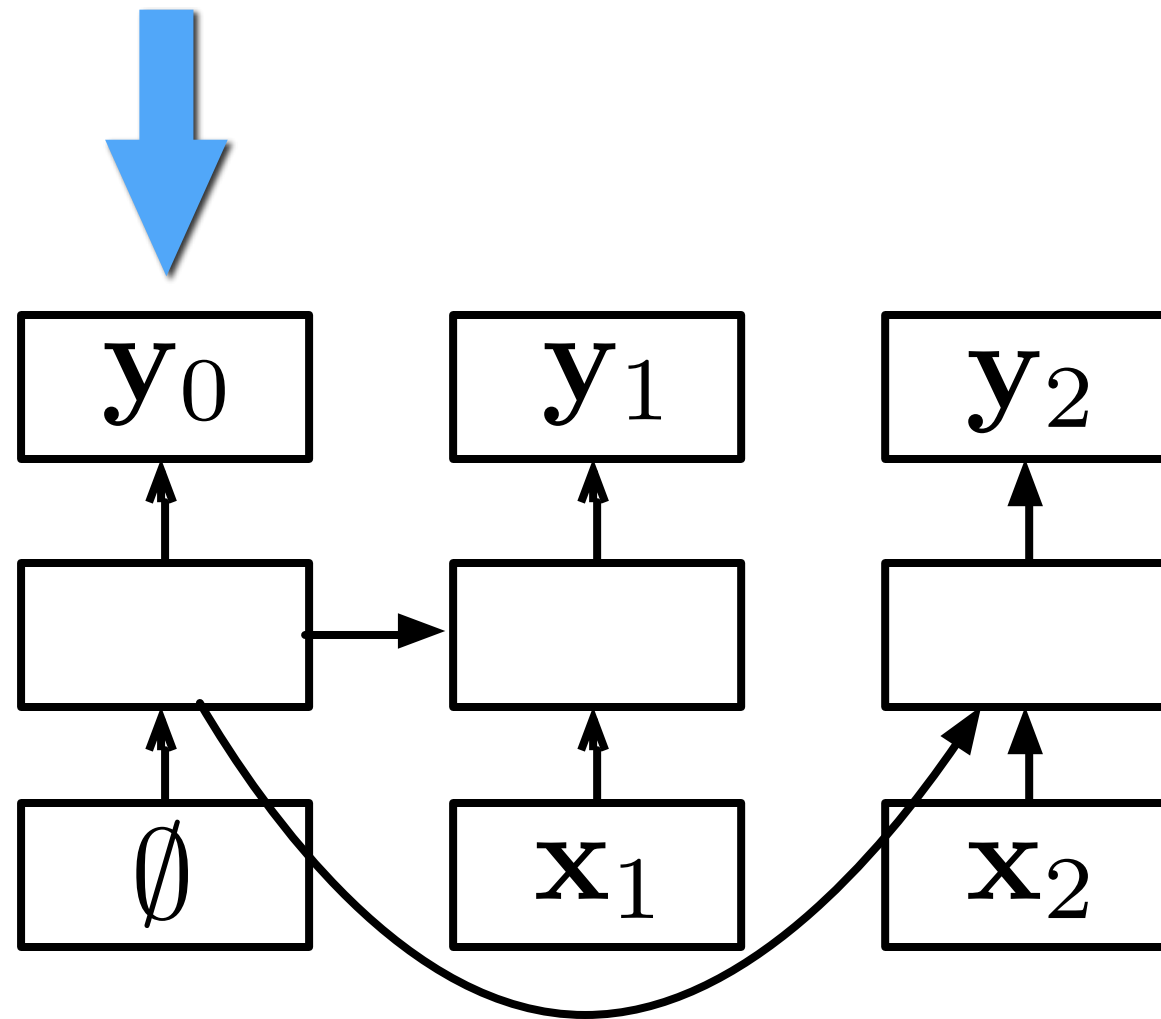
Implementing RNNs

Stack RNNs



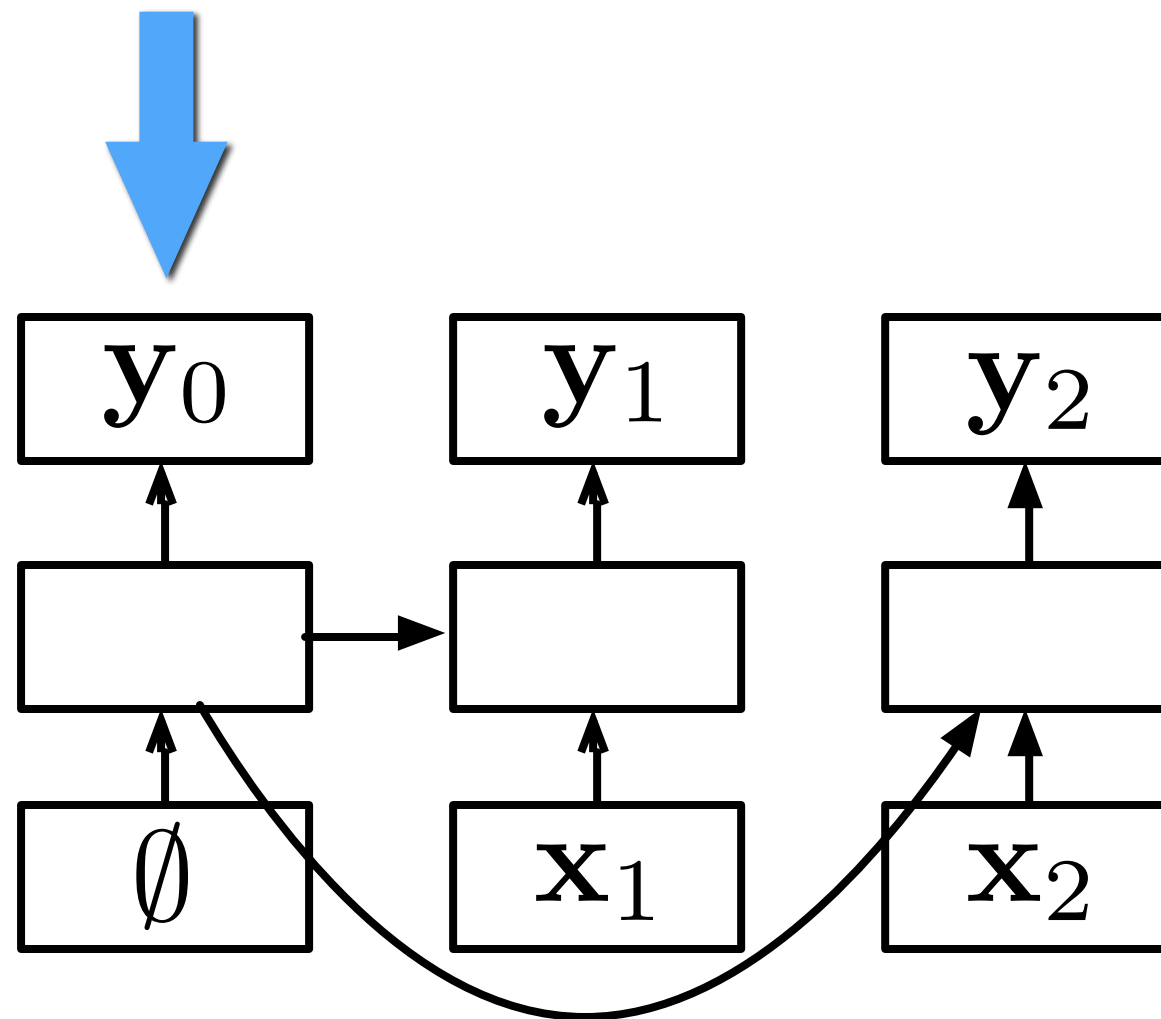
Implementing RNNs

Stack RNNs



Implementing RNNs

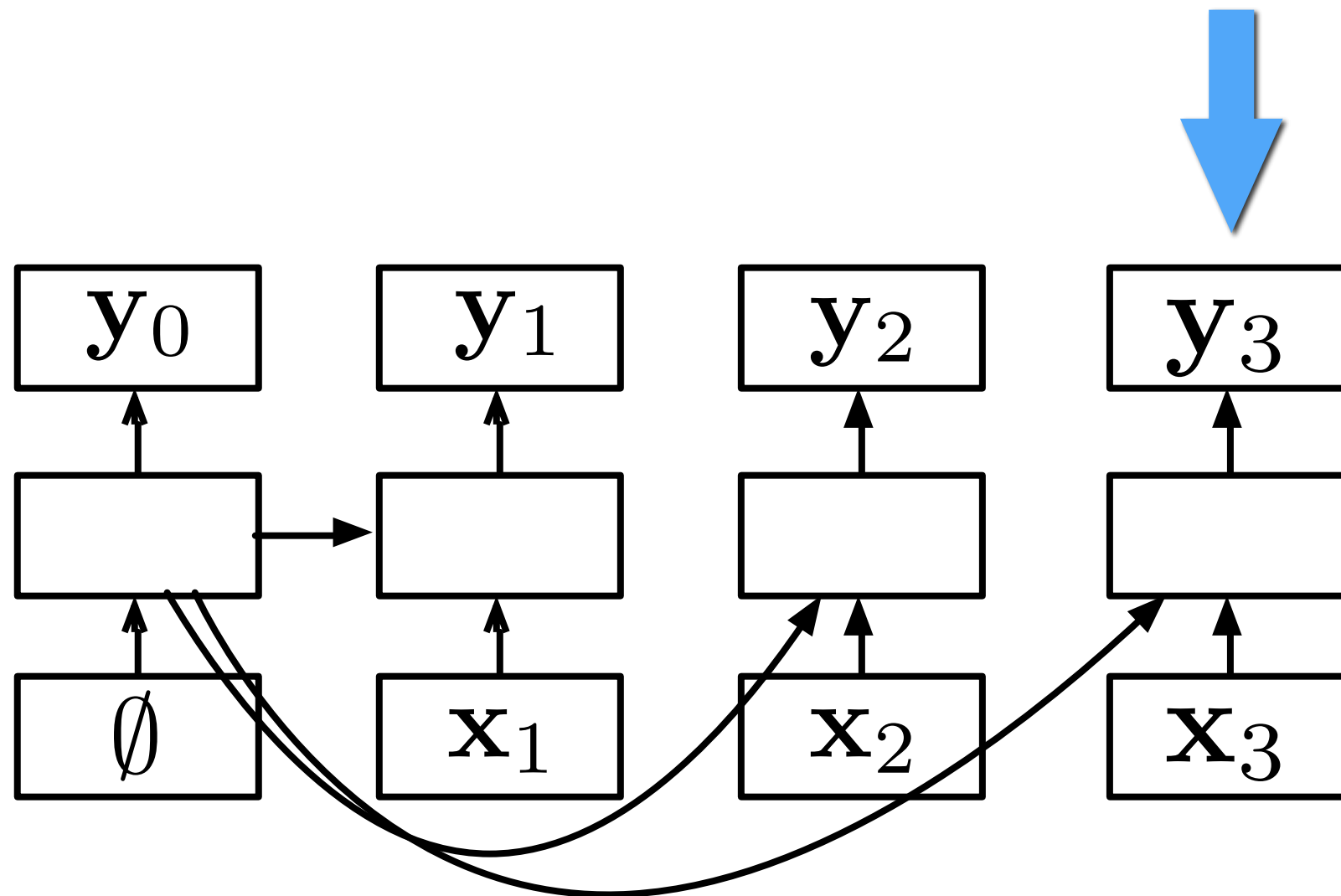
Stack RNNs



PUSH

Implementing RNNs

Stack RNNs



Importance Sampling

Assume we've got a conditional distribution $q(\mathbf{y} \mid \mathbf{x})$

- s.t.
- (i) $p(\mathbf{x}, \mathbf{y}) > 0 \implies q(\mathbf{y} \mid \mathbf{x}) > 0$
 - (ii) $\mathbf{y} \sim q(\mathbf{y} \mid \mathbf{x})$ is tractable and
 - (iii) $q(\mathbf{y} \mid \mathbf{x})$ is tractable

Importance Sampling

Assume we've got a conditional distribution $q(\mathbf{y} \mid \mathbf{x})$

- s.t.
- (i) $p(\mathbf{x}, \mathbf{y}) > 0 \implies q(\mathbf{y} \mid \mathbf{x}) > 0$
 - (ii) $\mathbf{y} \sim q(\mathbf{y} \mid \mathbf{x})$ is tractable and
 - (iii) $q(\mathbf{y} \mid \mathbf{x})$ is tractable

Let the importance weights $w(\mathbf{x}, \mathbf{y}) = \frac{p(\mathbf{x}, \mathbf{y})}{q(\mathbf{y} \mid \mathbf{x})}$

Importance Sampling

Assume we've got a conditional distribution $q(\mathbf{y} \mid \mathbf{x})$

- s.t.
- (i) $p(\mathbf{x}, \mathbf{y}) > 0 \implies q(\mathbf{y} \mid \mathbf{x}) > 0$
 - (ii) $\mathbf{y} \sim q(\mathbf{y} \mid \mathbf{x})$ is tractable and
 - (iii) $q(\mathbf{y} \mid \mathbf{x})$ is tractable

Let the importance weights $w(\mathbf{x}, \mathbf{y}) = \frac{p(\mathbf{x}, \mathbf{y})}{q(\mathbf{y} \mid \mathbf{x})}$

$$\begin{aligned} p(\mathbf{x}) &= \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x})} p(\mathbf{x}, \mathbf{y}) = \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x})} w(\mathbf{x}, \mathbf{y}) q(\mathbf{y} \mid \mathbf{x}) \\ &= \mathbb{E}_{\mathbf{y} \sim q(\mathbf{y} \mid \mathbf{x})} w(\mathbf{x}, \mathbf{y}) \end{aligned}$$

Importance Sampling

$$\begin{aligned} p(\boldsymbol{x}) &= \sum_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})} p(\boldsymbol{x}, \boldsymbol{y}) = \sum_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})} w(\boldsymbol{x}, \boldsymbol{y}) q(\boldsymbol{y} \mid \boldsymbol{x}) \\ &= \mathbb{E}_{\boldsymbol{y} \sim q(\boldsymbol{y} \mid \boldsymbol{x})} w(\boldsymbol{x}, \boldsymbol{y}) \end{aligned}$$

Importance Sampling

$$\begin{aligned} p(\boldsymbol{x}) &= \sum_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})} p(\boldsymbol{x}, \boldsymbol{y}) = \sum_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})} w(\boldsymbol{x}, \boldsymbol{y}) q(\boldsymbol{y} \mid \boldsymbol{x}) \\ &= \mathbb{E}_{\boldsymbol{y} \sim q(\boldsymbol{y} \mid \boldsymbol{x})} w(\boldsymbol{x}, \boldsymbol{y}) \end{aligned}$$

Replace this expectation with its Monte Carlo estimate.

$$\boldsymbol{y}^{(i)} \sim q(\boldsymbol{y} \mid \boldsymbol{x}) \quad \text{for } i \in \{1, 2, \dots, N\}$$

Importance Sampling

$$\begin{aligned} p(\boldsymbol{x}) &= \sum_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})} p(\boldsymbol{x}, \boldsymbol{y}) = \sum_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})} w(\boldsymbol{x}, \boldsymbol{y}) q(\boldsymbol{y} \mid \boldsymbol{x}) \\ &= \mathbb{E}_{\boldsymbol{y} \sim q(\boldsymbol{y} \mid \boldsymbol{x})} w(\boldsymbol{x}, \boldsymbol{y}) \end{aligned}$$

Replace this expectation with its Monte Carlo estimate.

$$\boldsymbol{y}^{(i)} \sim q(\boldsymbol{y} \mid \boldsymbol{x}) \quad \text{for } i \in \{1, 2, \dots, N\}$$

$$\mathbb{E}_{q(\boldsymbol{y} \mid \boldsymbol{x})} w(\boldsymbol{x}, \boldsymbol{y}) \stackrel{\text{MC}}{\approx} \frac{1}{N} \sum_{i=1}^N w(\boldsymbol{x}, \boldsymbol{y}^{(i)})$$