Artificial neural networks and the challenge of compositional generalization

Marco Baroni

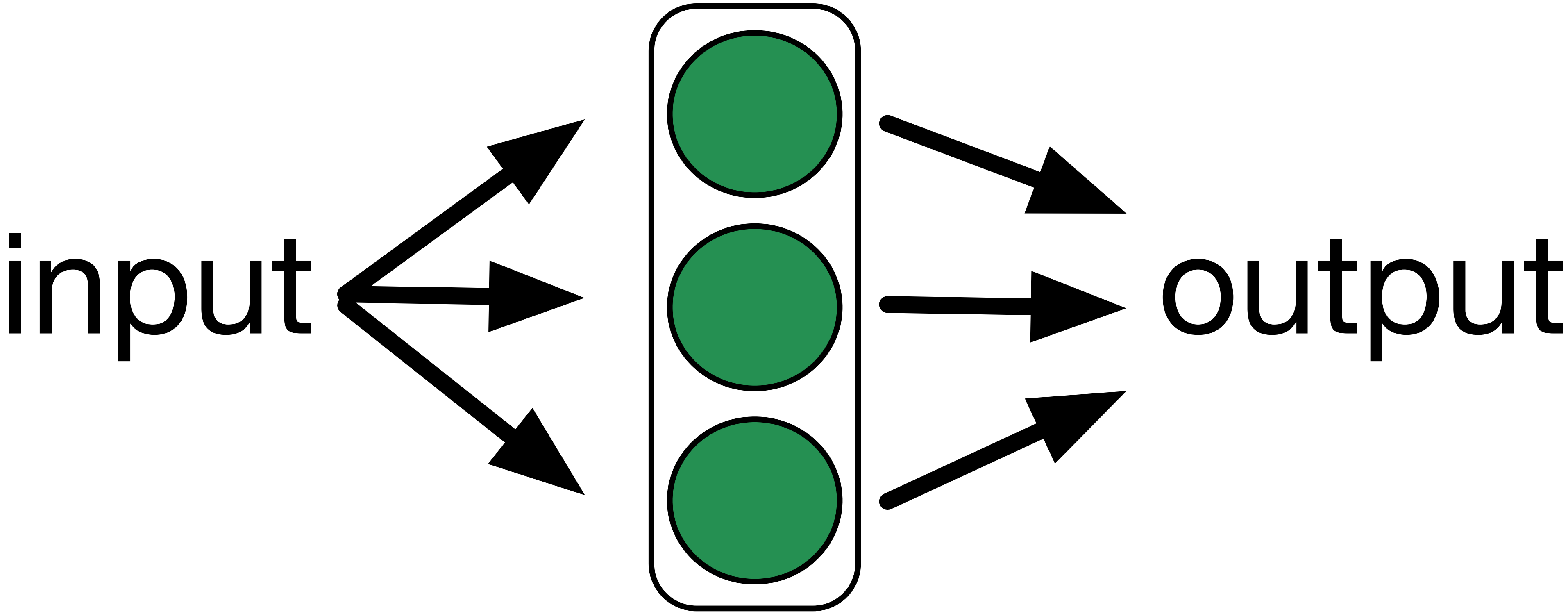
Facebook AI Research

Outline

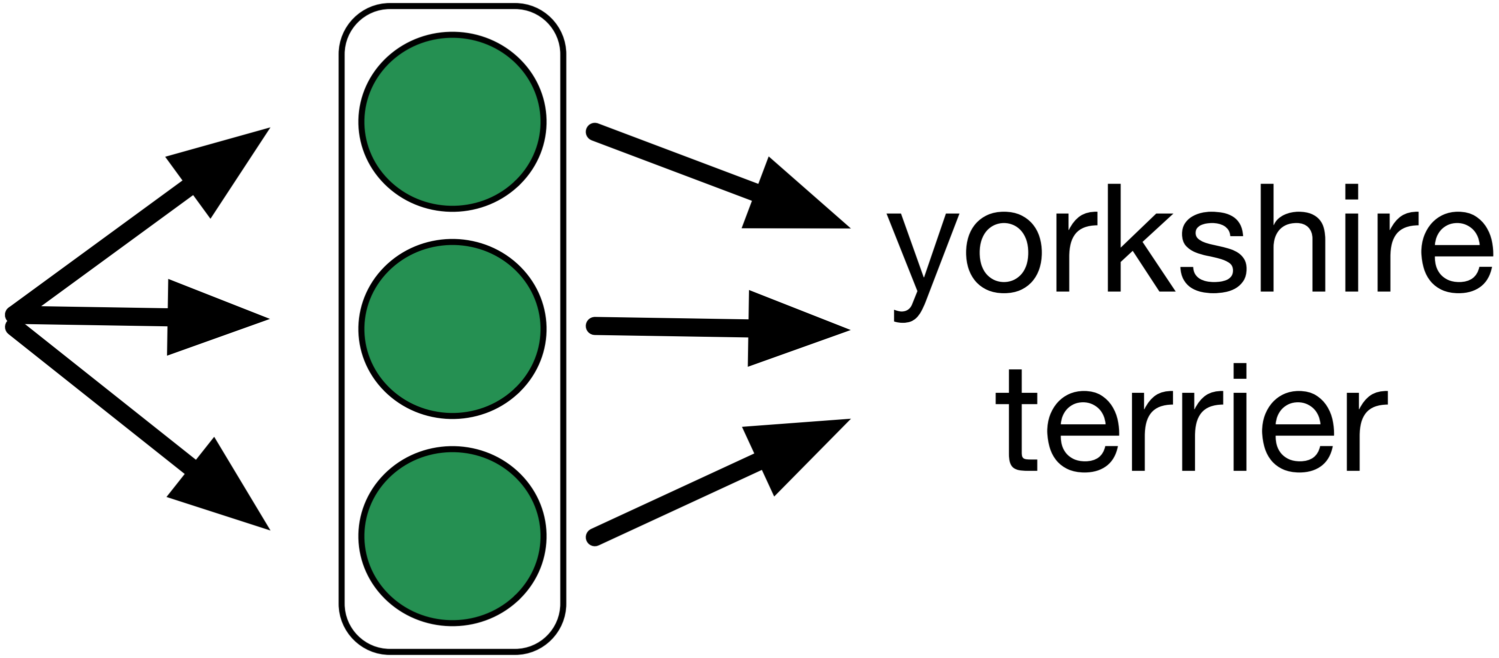
- Recurrent neural networks
- A compositional challenge for neural networks (and humans)
- (If time allows) Looking for a compositional neural network in a haystack

Caution: This is the "bad cop" talk

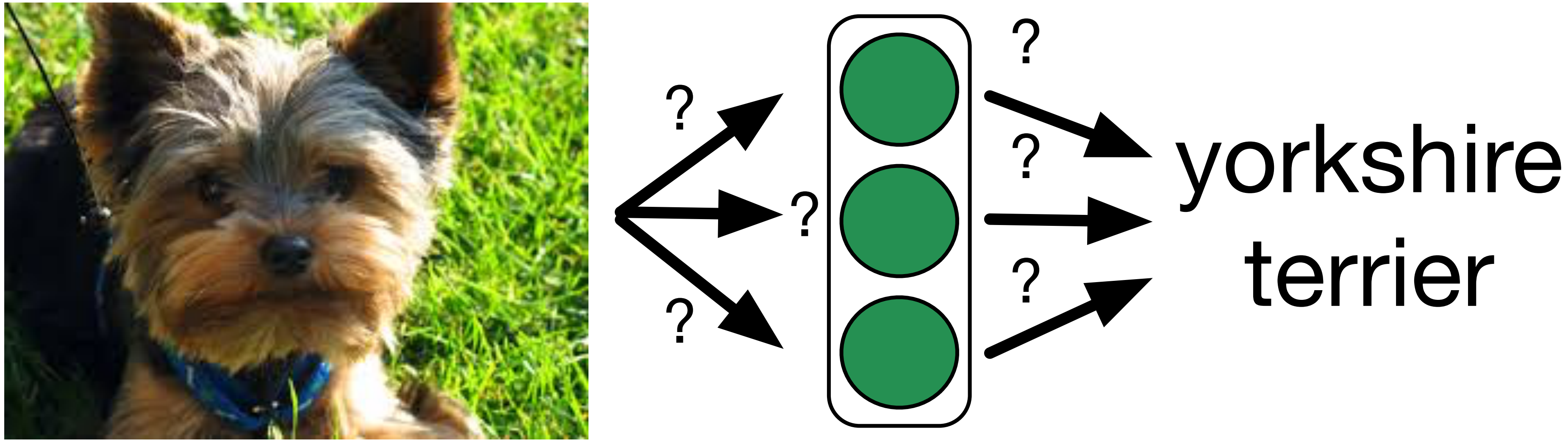
Artificial neural networks



Artificial neural networks



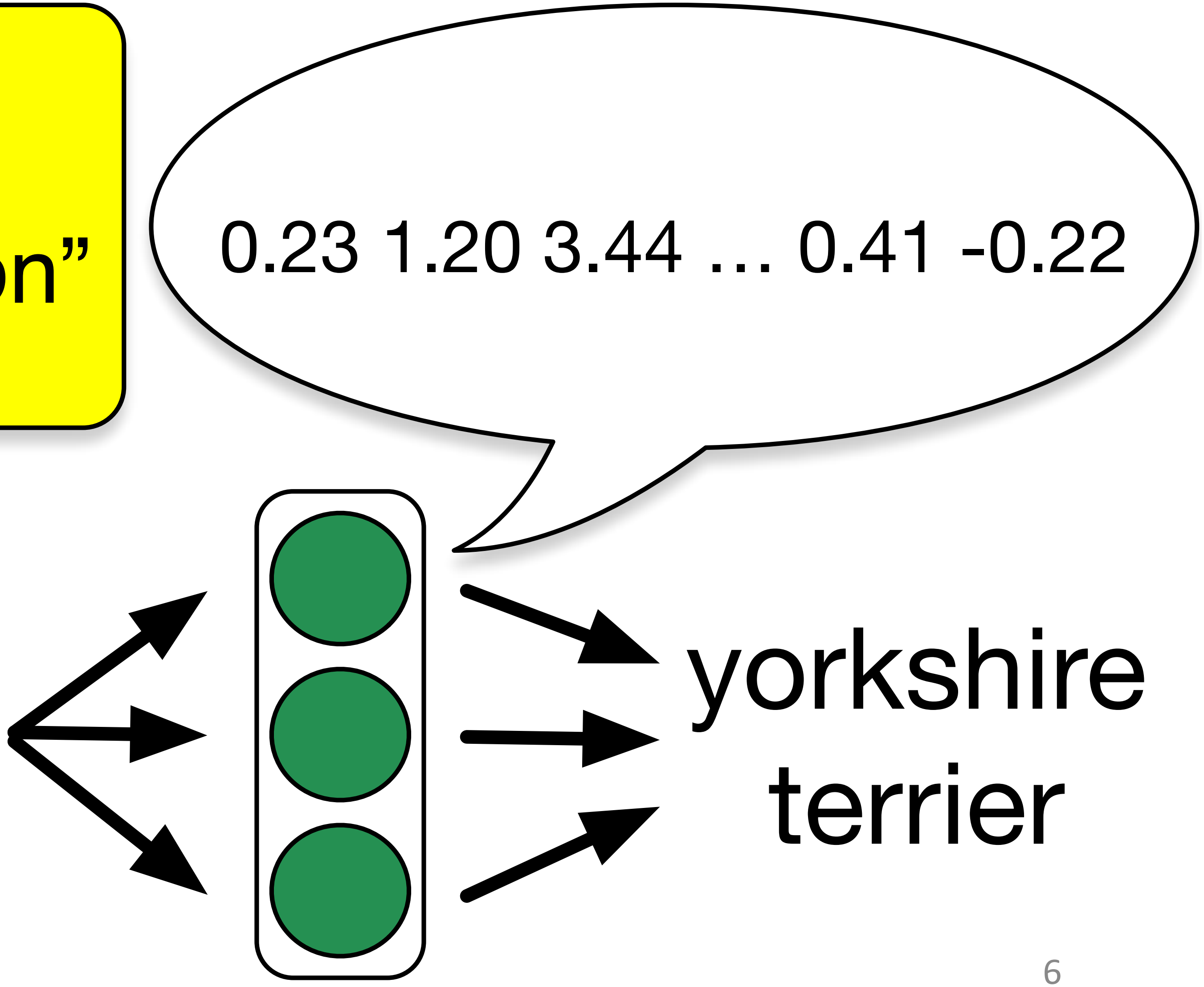
Artificial neural networks



"training" consists in optimally setting network weights to produce right output for each example input

Artificial neural networks

network automatically
produces its own
“distributed representation”
of the input

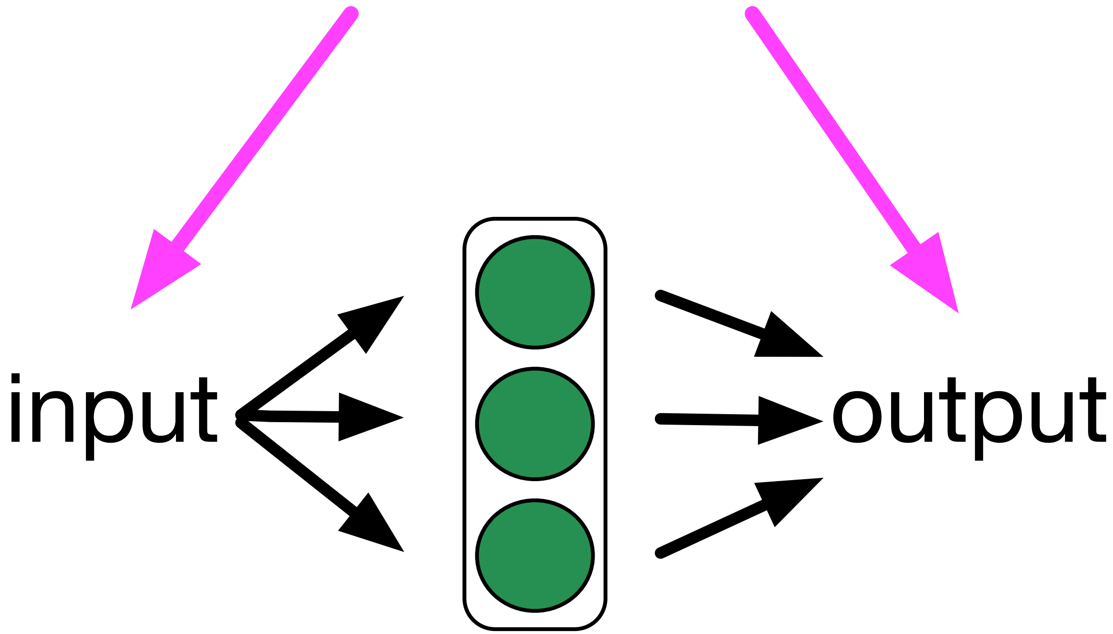


The generality of neural networks

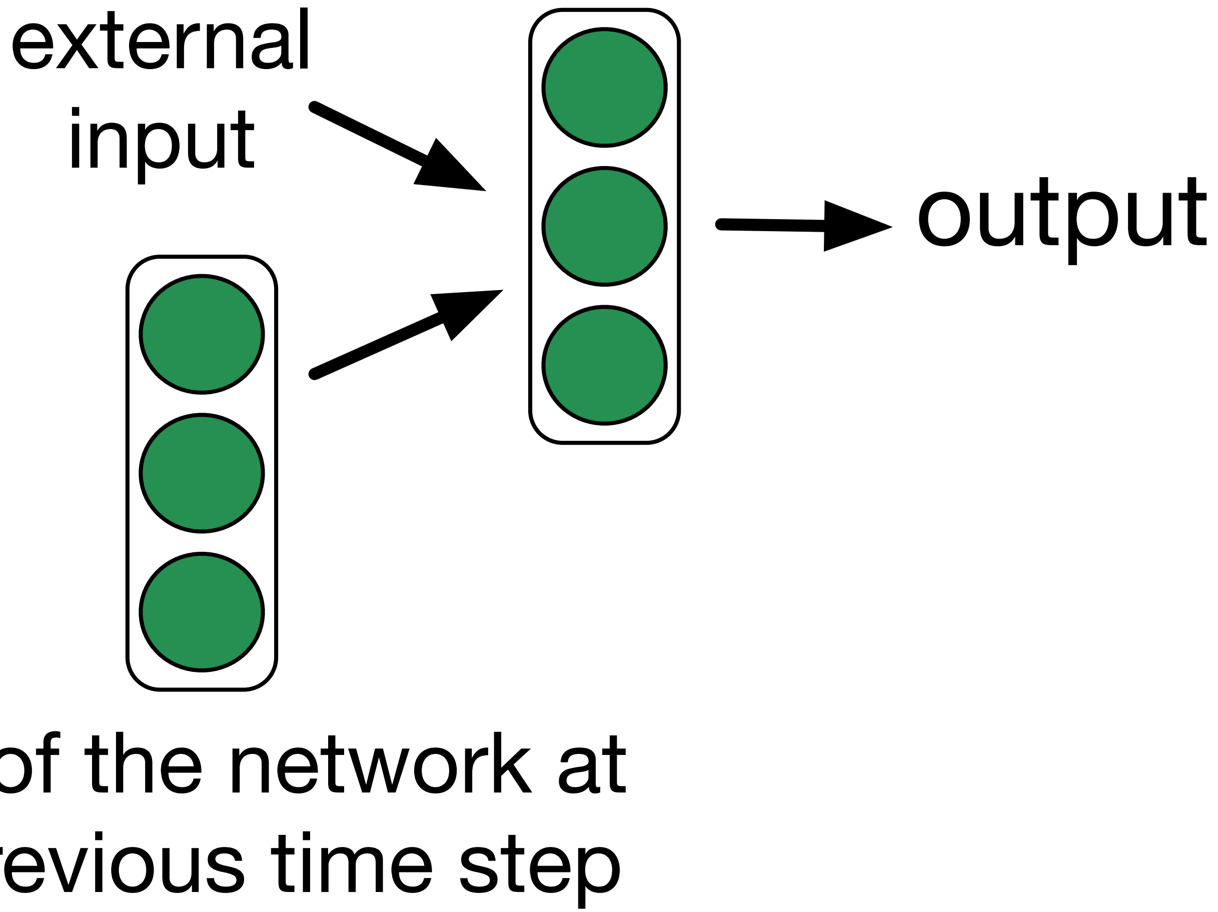
I: images, O: object labels
I: documents, O: topics
I: pictures of cars, O: voting preferences

…

training agnostic to nature of input and output

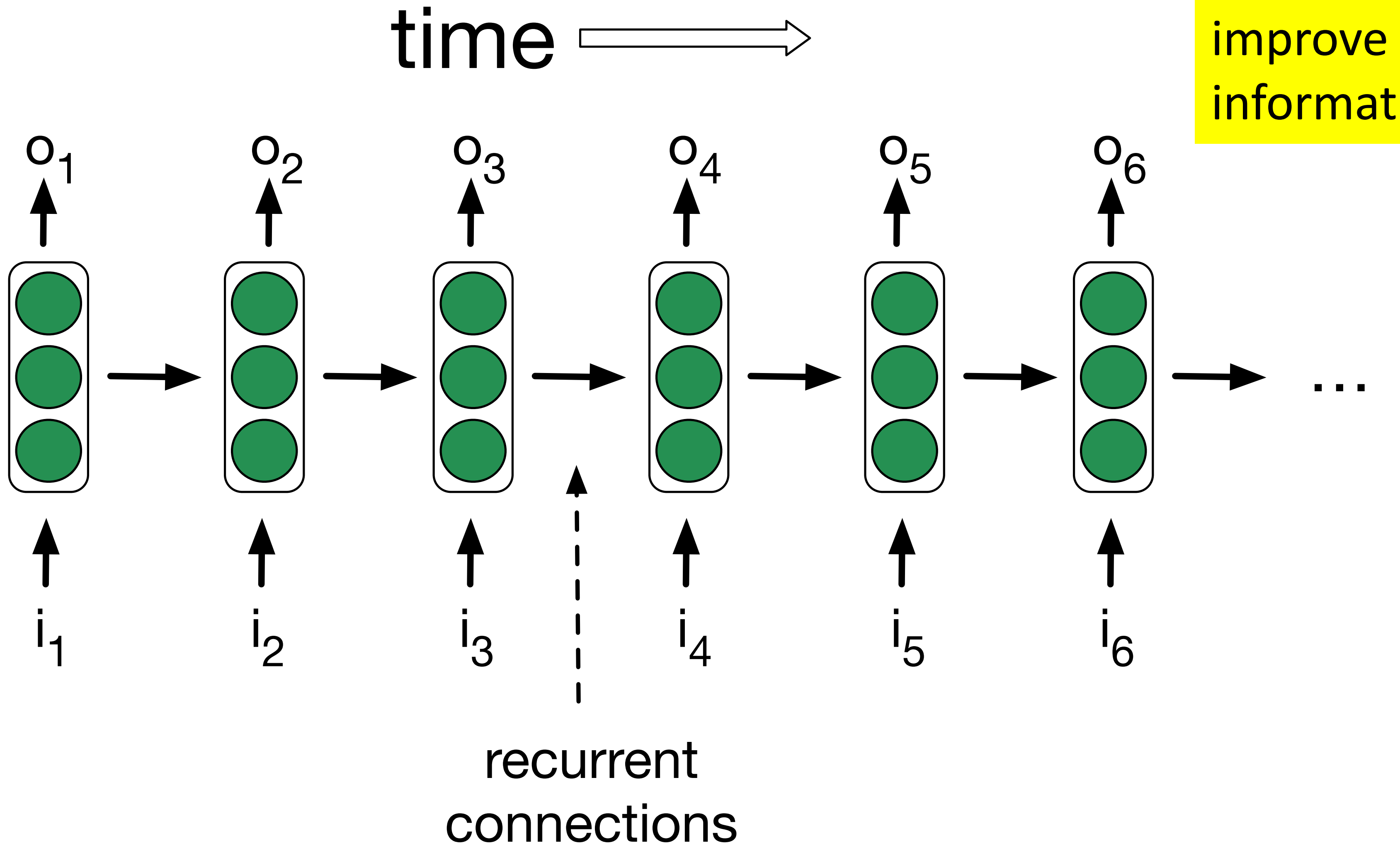


Taking time into account with recurrent connections



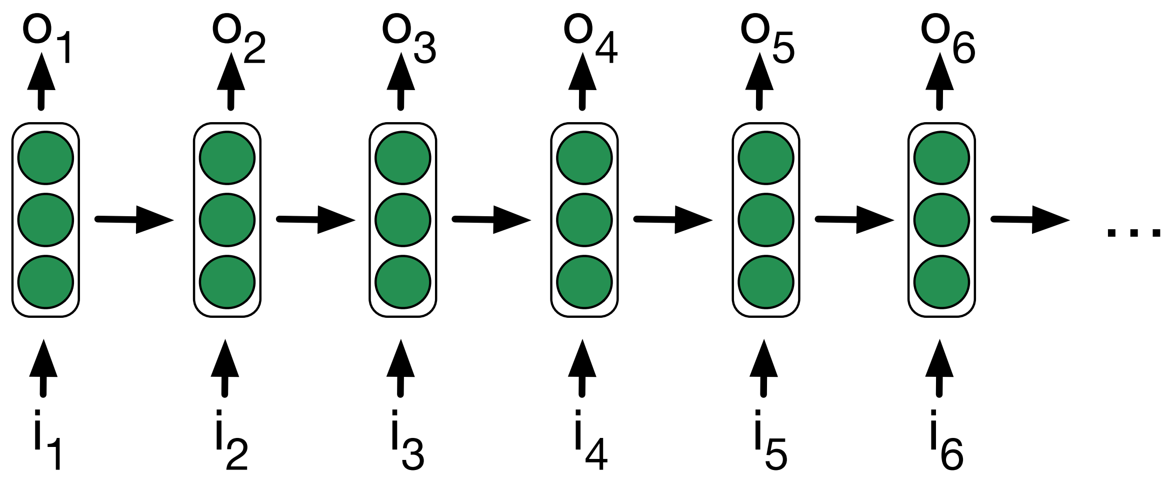
Recurrent neural networks
The "unfolded" view

Modern RNNs (e.g., LSTMs) possess gating mechanism that improve temporal information flow

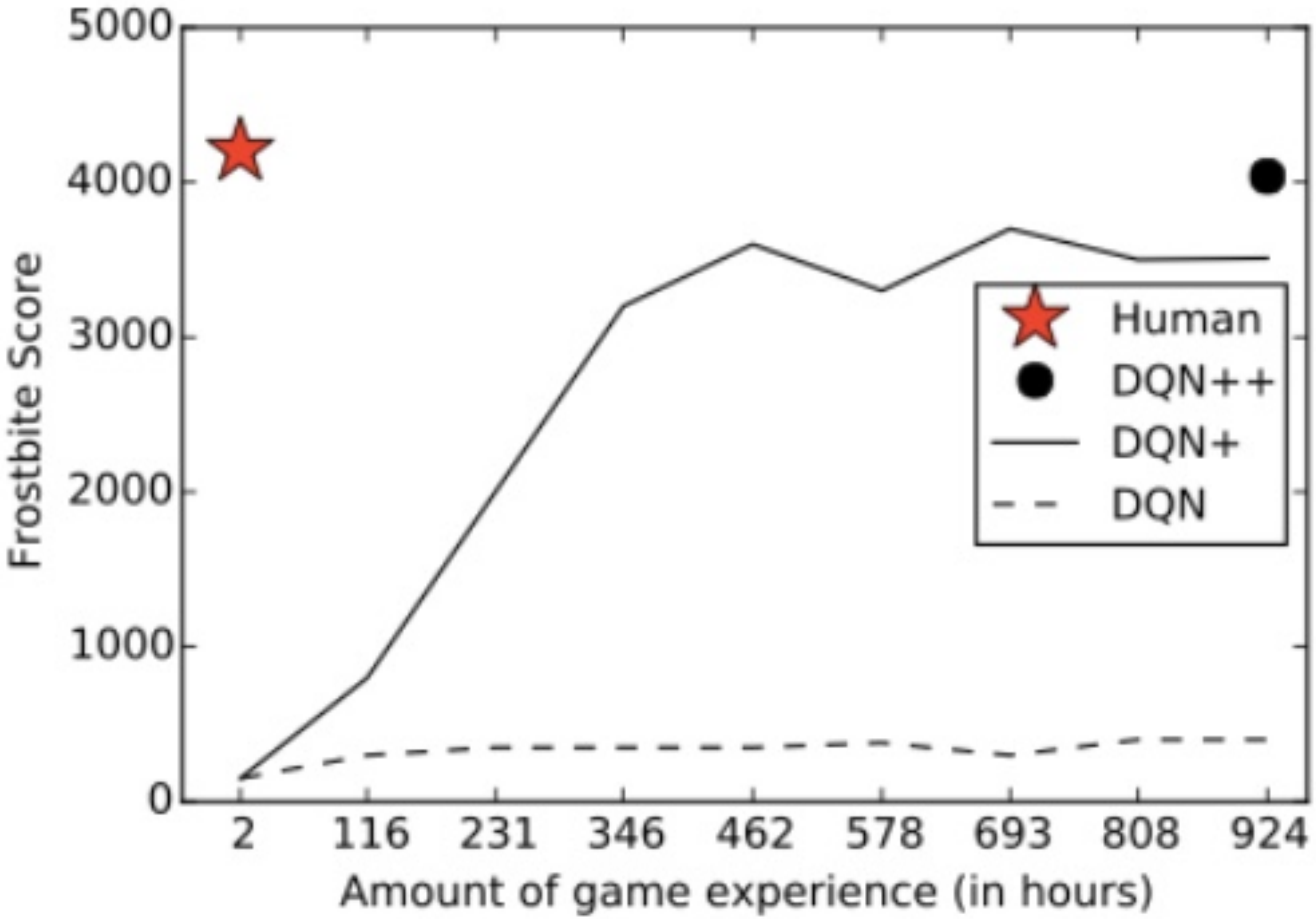


The generality of recurrent neural networks

I: English sentences, O: French sentences
I: linguistic instructions, O: action sequences
I: video game states, O: next actions
…



Are we on the verge of general machine intelligence?



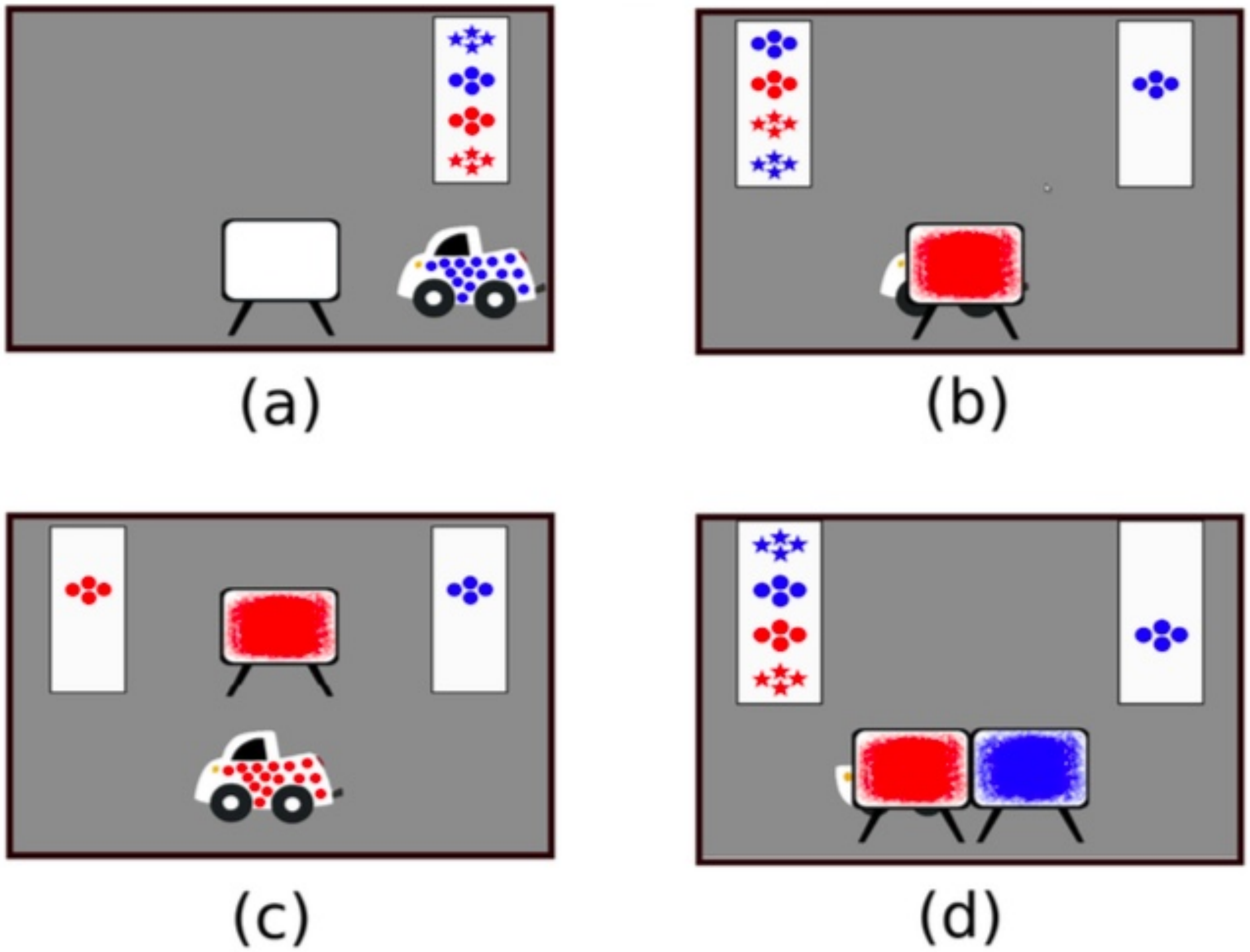
Lake et al. 2018

When are we humans fast at learning?

- When evolution has done the slow learning work for us
  - Perception and categorization, naïve physics and psychology, motor skills, core language faculties, reasoning...

- When new problems can be solved by combining old tricks (compositionality)

Compositional reasoning in 4-year olds



Piantadosi and Aslin 2016

Outline

- Recurrent neural networks
- **A compositional challenge for neural networks (and humans)**
- (If time allows) Looking for a compositional neural network in a haystack

- Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. ICML 2018

- The SCAN challenge: https://github.com/brendenlake/SCAN/

Lots of earlier work on neural networks and systematicity, main novelty here is that we test latest-generation, state-of-the-art architectures!

Systematic compositionality

Fodor and Pylyshyn 1988, Marcus 2003, 2018...

- Walk
- Walk twice
- Run
- Run twice

Systematic compositionality

Fodor and Pylyshyn 1988, Marcus 2003, 2018...

- Walk
- Walk twice
- Run
- Run twice
- Dax

Systematic compositionality

Fodor and Pylyshyn 1988, Marcus 2003, 2018...

- Walk
- Walk twice
- Run
- Run twice
- Dax
- Dax twice

Systematic compositionality

Fodor and Pylyshyn 1988, Marcus 2003, 2018...

- Walk
- Walk twice
- Run
- Run twice
- Dax
- Dax twice

[[X twice]] = [[X]][[X]]

[[dax]] = perform daxing action

... or perhaps meanings include algorithmic components such as:
for (c=0,c<3,c++) {perform X}

Systematic compositionality in a simple grounded environment

walk and turn left!

Testing generalization

TRAINING PHASE

walk
WALK

jump after walk
WALK JUMP

walk and jump left
WALK LTURN JUMP

run thrice
RUN RUN RUN

run around right
RTURN RUN RTURN RUN
RTURN RUN RTURN RUN

look right and
walk left
RTURN LOOK
LTURN WALK

walk and run
RUN WALK

TEST TIME

jump around
and run

The SCAN commands: examples

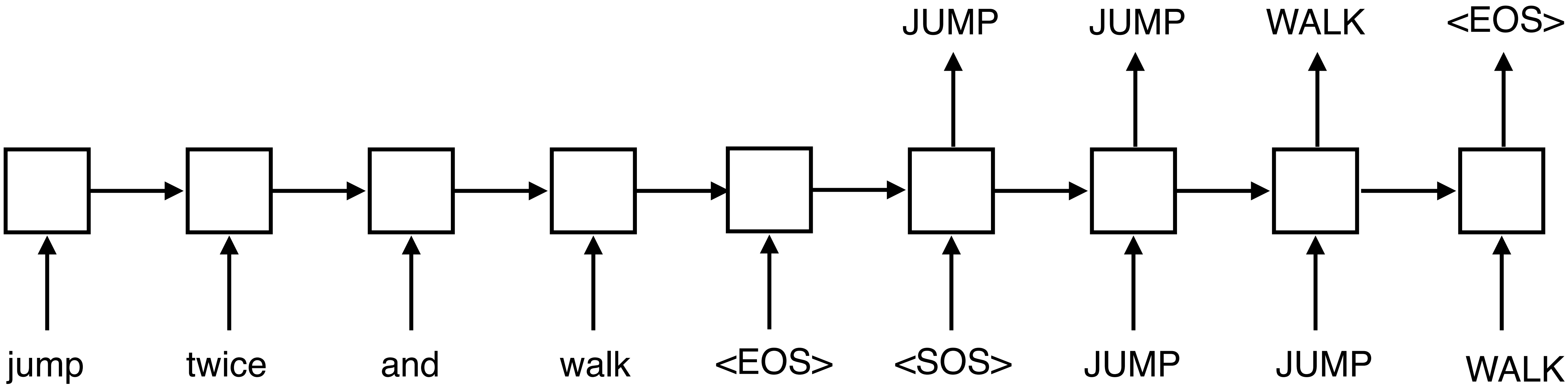
- Primitive commands:
  - run -> RUN
  - walk -> WALK
  - turn left -> LTURN

- Modifiers:
  - walk left -> LTURN WALK
  - run twice -> RUN RUN

- Conjunctions:
  - walk left and run twice -> LTURN WALK RUN RUN
  - run twice after walk left -> RUN RUN LTURN WALK

- Simplifications:
  - No scope ambiguity ("walk and [run twice]")
  - No recursion ("walk and run" vs *"walk and run and walk")

Sequence-to-sequence RNNs for SCAN



General methodology

- Train sequence-to-sequence RNN on 100k commands and corresponding action sequences
- At test time, only *new* composed commands presented
- Each test command presented once
- RNN must generate right action sequence at first try

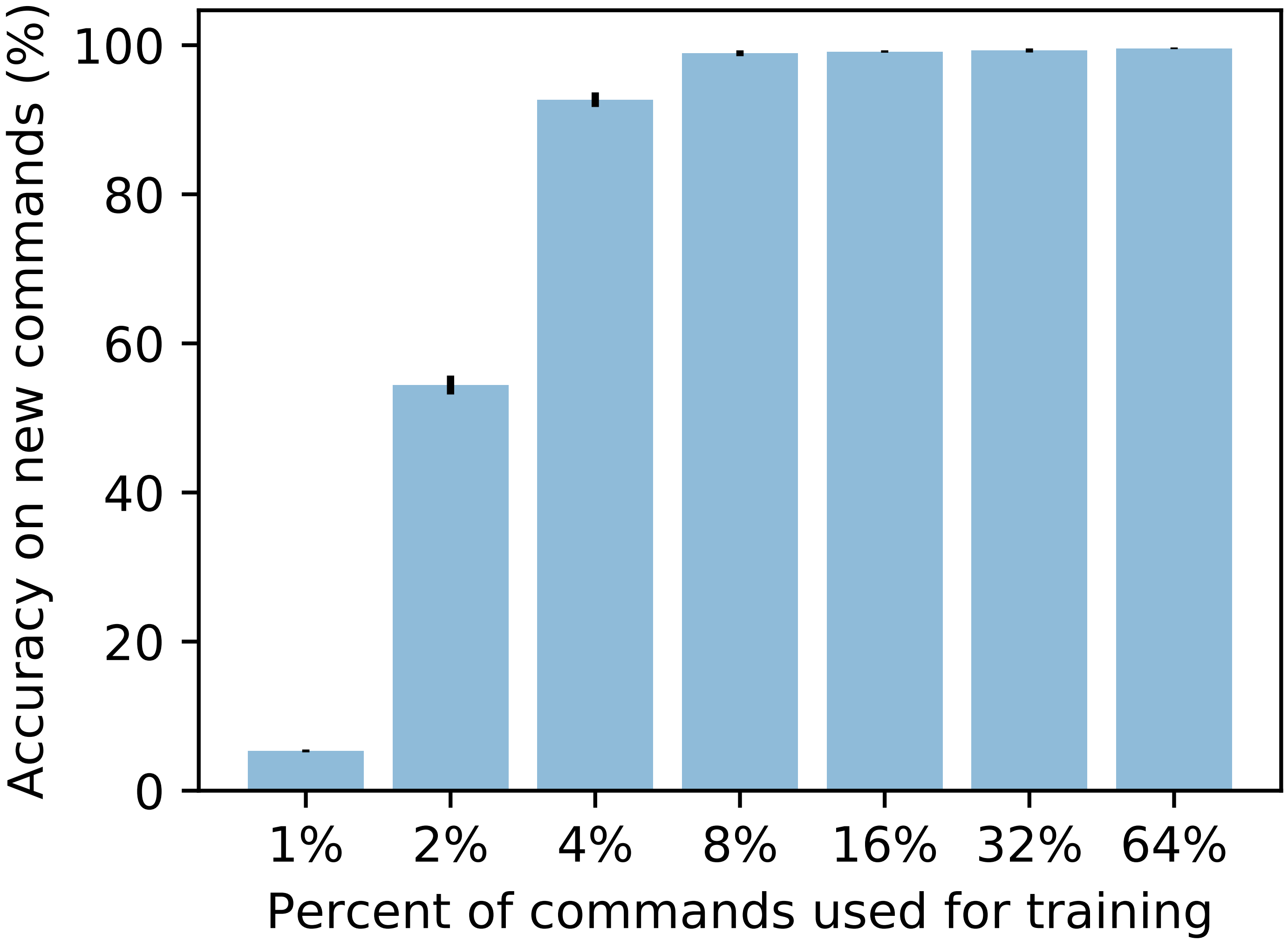
- Training details: ADAM optimization with 0.001 learning rate and 50% teacher forcing
- Best model overall:
  - 2-layer LSTM with 200 hidden units per layer, no attention, 0.5 dropout

Experiment 1: random train/test split

- Included in training tasks:
  - look around left twice
  - look around left twice and turn left
  - jump right twice
  - run twice and jump right twice

- Presented during testing:
  - look around left twice and jump right twice

Random train/test split results



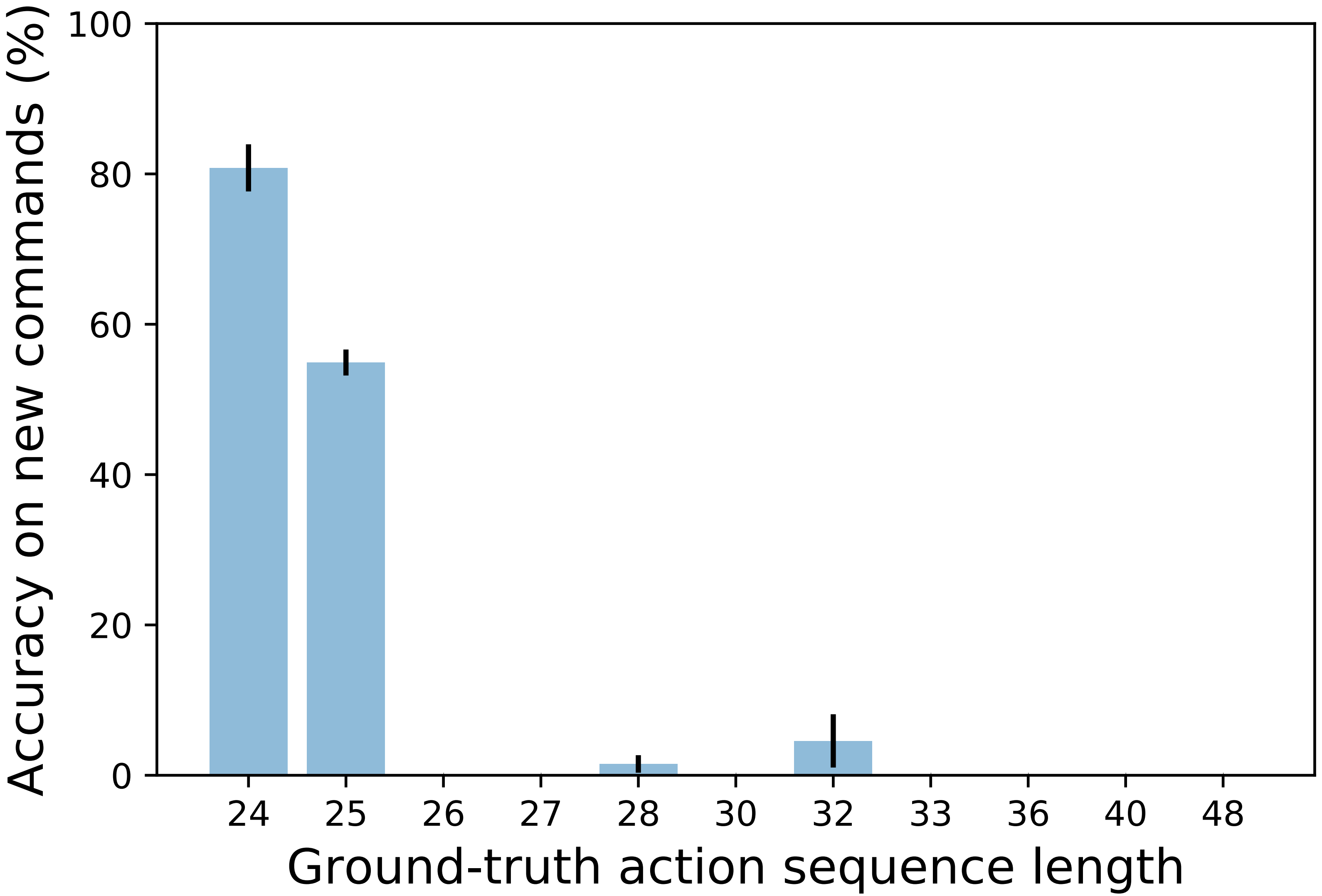
Experiment 2: split by action length

A grammar must reflect and explain the ability of a speaker to produce and understand new sentences which may be longer than any he has previously heard (Chomsky 1956)

- Train on commands requiring shorter action sequences (up to 22 actions)
  - jump around left twice (16 actions)
  - walk opposite right thrice (9 actions)
  - jump around left twice and walk opposite right twice (22 actions)

- Test on commands requiring longer actions sequences (from 24 to 48 actions)
  - jump around left twice and walk opposite right thrice (25 actions)

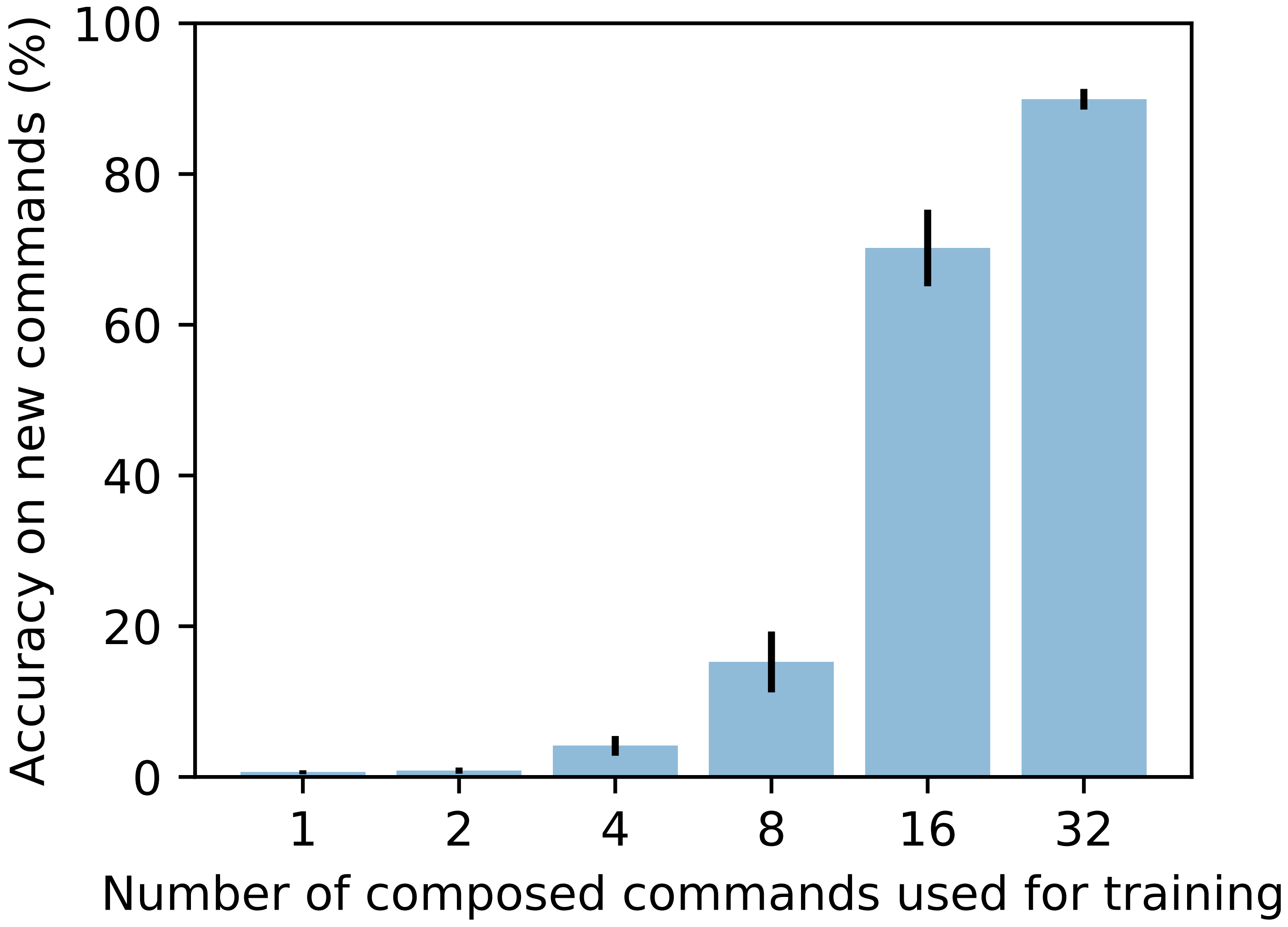
Length split results



Experiment 3: generalizing composition of a primitive command (the "dax" experiment)

- Training set contains all possible commands with "run", "walk", look", "turn left", "turn right":
  - "run", "run twice", "turn left and run opposite thrice", "walk after run", ...
- *but only a small set of composed "jump" commands:*
  - "jump", "jump left", "run and jump", "jump around twice"
- System tested on all remaining "jump" commands:
  - jump twice
  - jump left and run opposite thrice
  - walk after jump
  - ...

Composed-"jump" split results



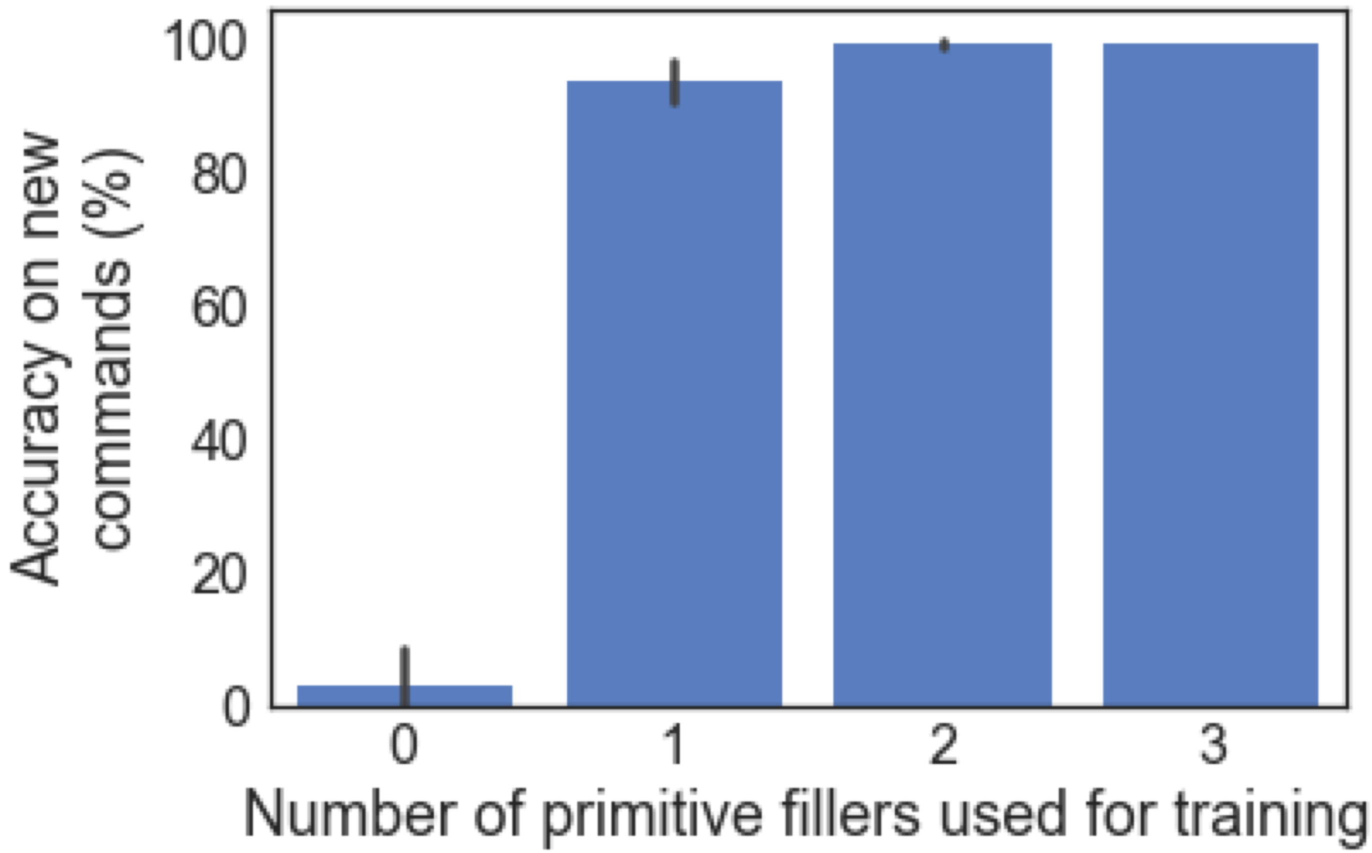
Experiment 4: generalizing the composition of familiar modifiers

- Training set includes all commands except those containing the *around right* combination:
  - "run", "run around left", "jump right and run around left thrice", "walk right after jump left", ...
- System tested on *around right* commands:
  - run around right
  - jump left and walk around right
  - ...

- Also less challenging splits in which all *X around right* commands are added to training set for 1, 2, 3 distinct fillers (verbs)

João Loula, Marco Baroni and Brenden Lake. Rearranging the familiar: Testing compositional generalization in recurrent networks. Blackbox WS at EMNLP 2018.

"Around right"-split results

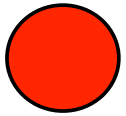
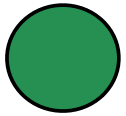


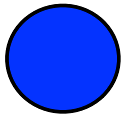
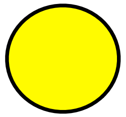
Ad-interim conclusion

- State-of-the-art "Seq2Seq" Recurrent Neural Networks achieve considerable degree of generalization (Exp 1)...
- ... but this generalization does not appear to be "systematically compositional" in the Fodorian sense (Exps 2-4)

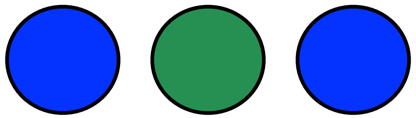
How do people dax twice?

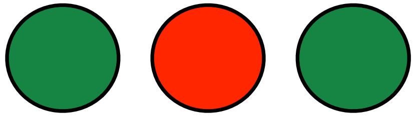
- Ongoing work with Brenden Lake and Tal Linzen

dax  blicket 

zup  tufa 

TRAINING

zup wif blicket 

blicket wif dax 

TEST

dax wif tufa

dax blicket

zup tufa

TRAINING

zup wif blicket

blicket wif dax

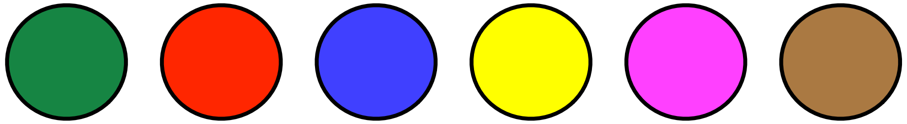
TEST

dax wif tufa

Lessons learned

- Average accuracies range from 88% to 65% for most difficult compositions (subject N $\cong$ 20)
- Subjects need to keep an eye on full training set while solving the task
- Systematic biases emerge in error patterns, studied in follow-up "blank state" experiments

"Blank state" experiments (subjects N = 29)

POOL: 

STIMULI:

fep

zup fep

fep dax fap

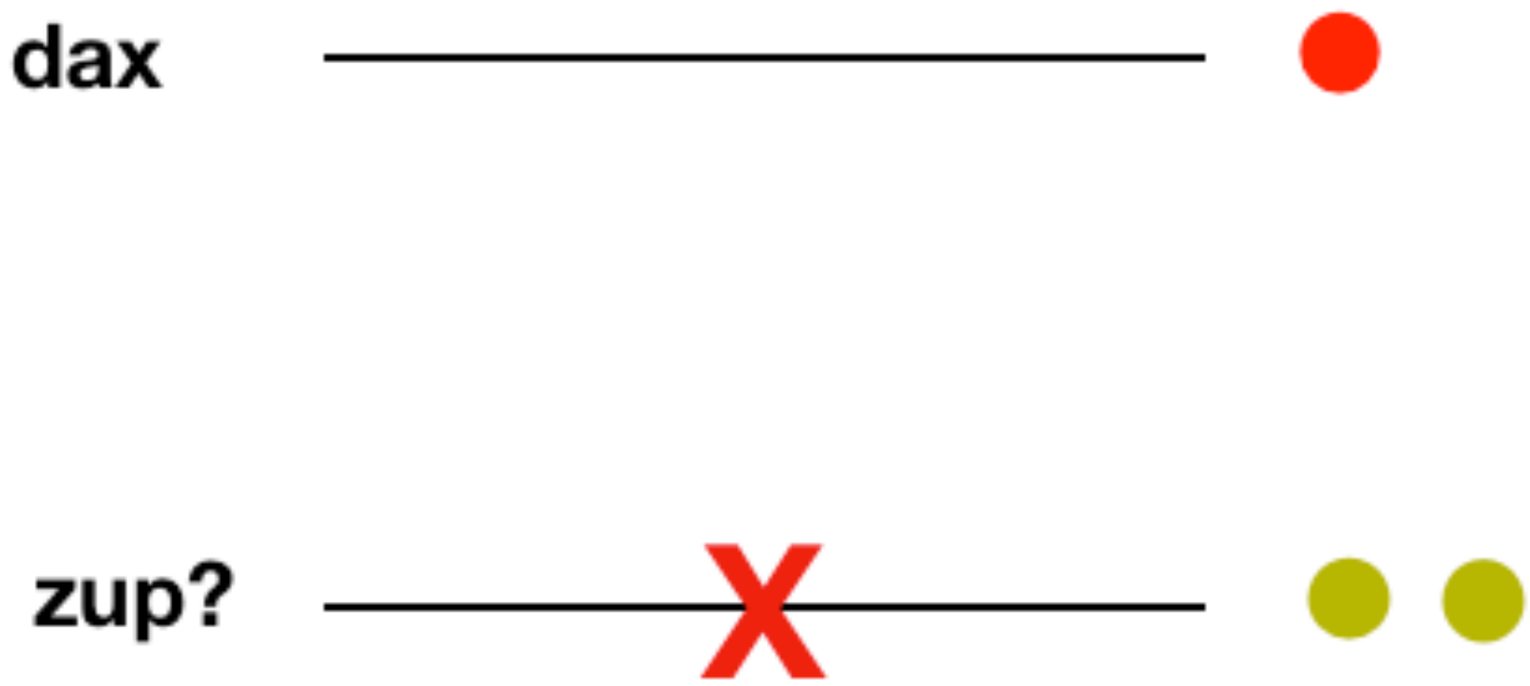
fep dax kiki

fep fep

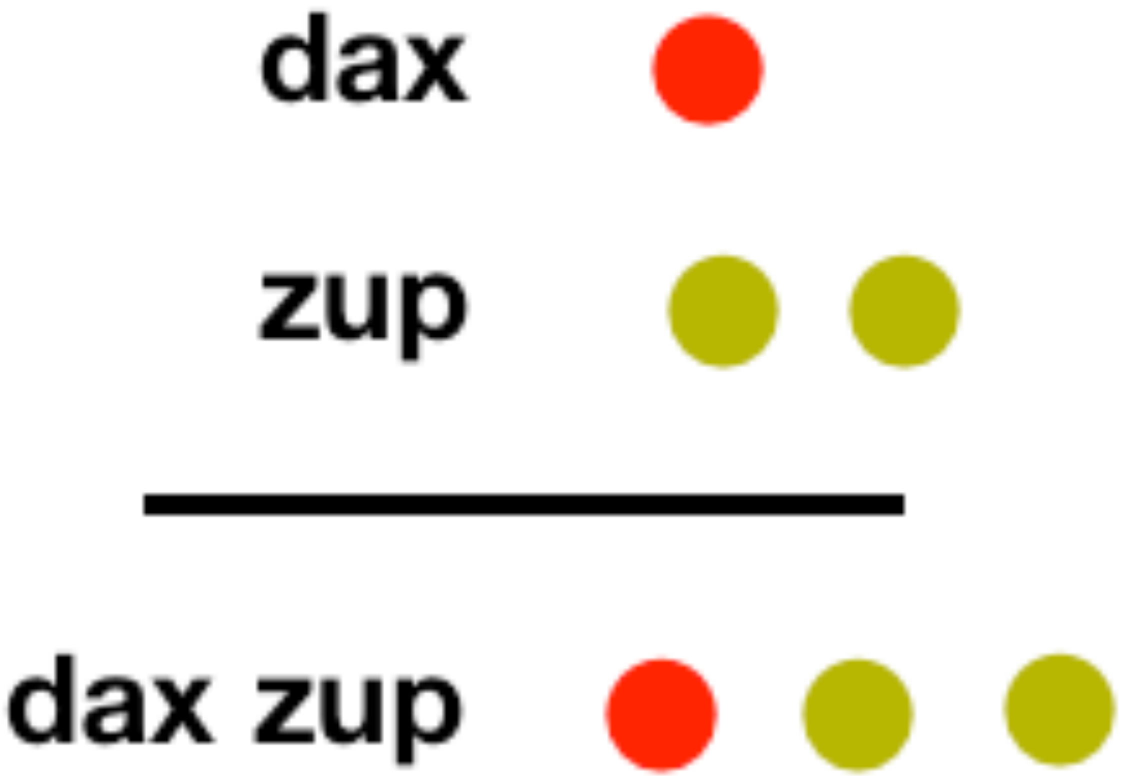
fep wif

kiki dax fep

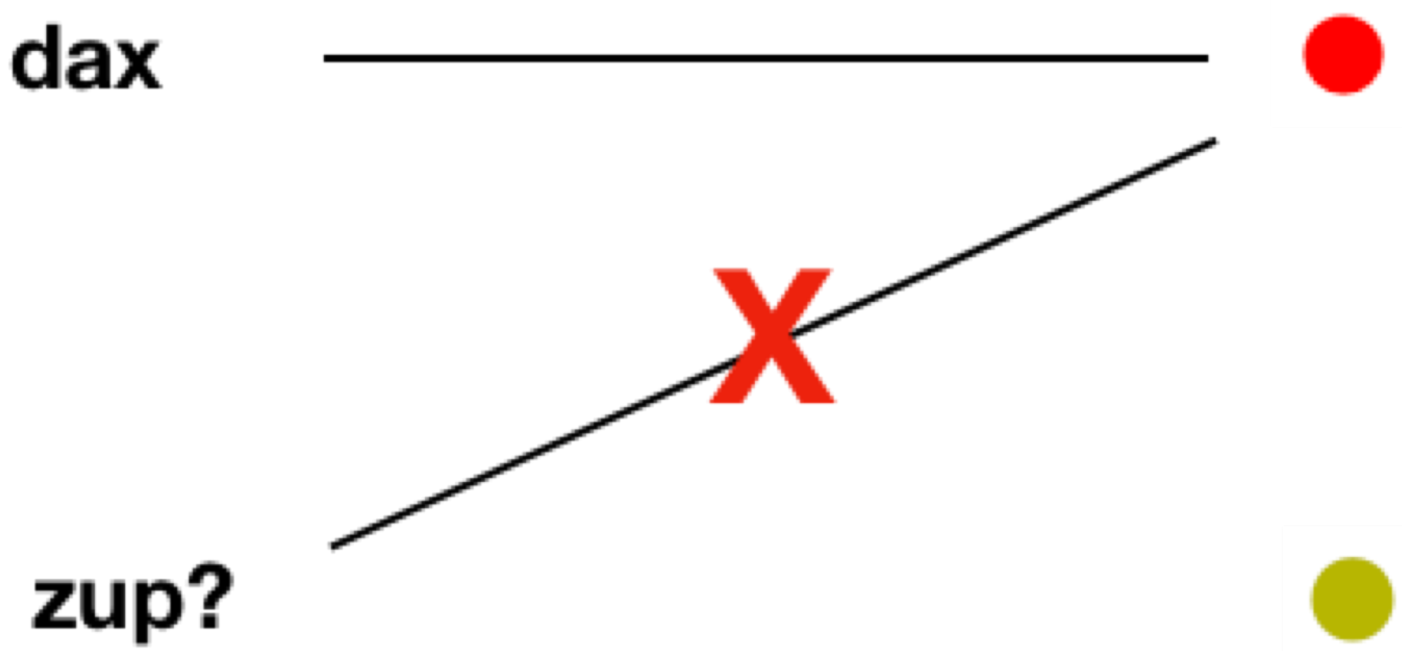
One-to-one mapping (62.1% of participants)



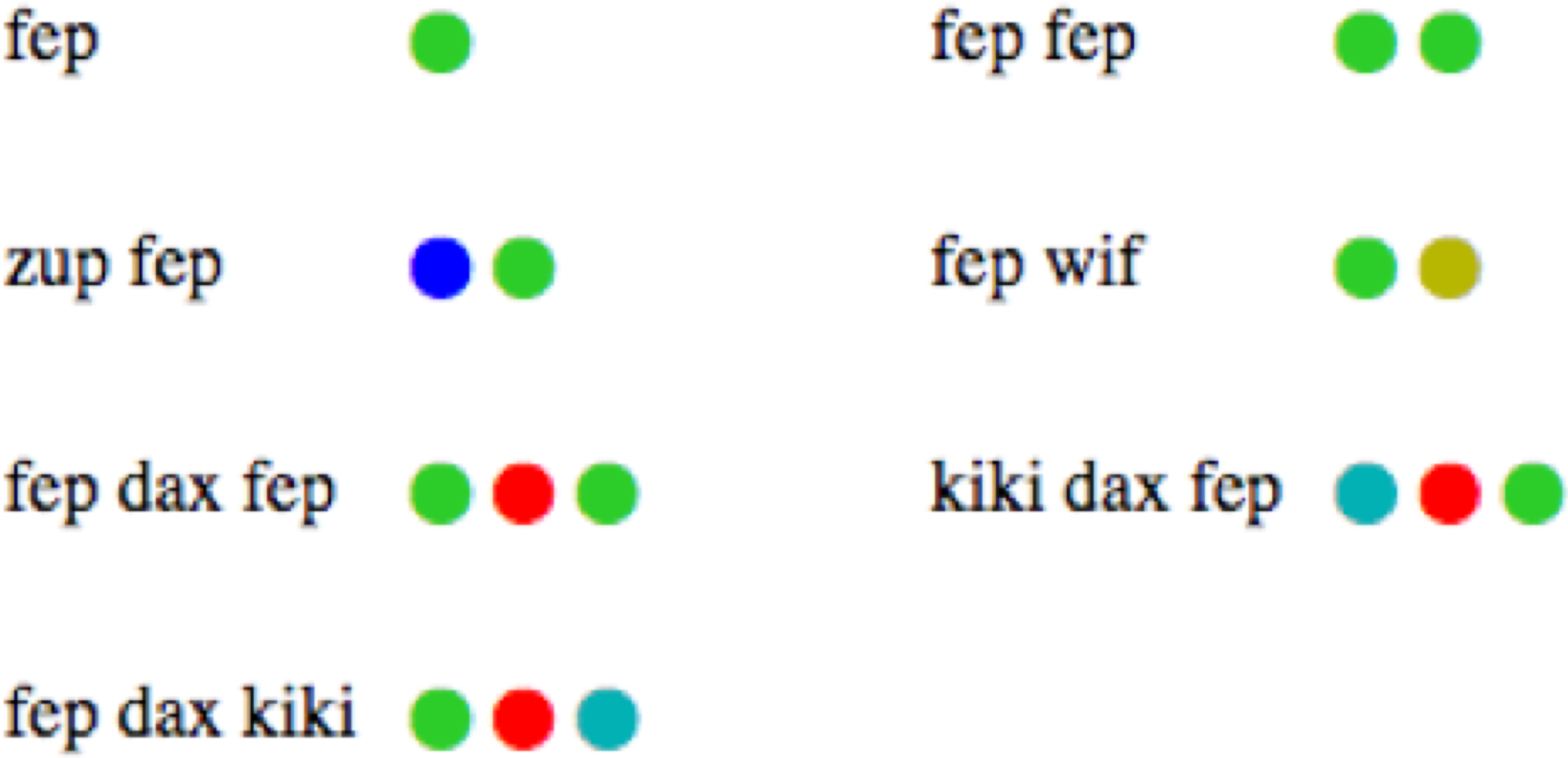
(Consistent) concatenation (79.3% of participants)



Mutual exclusivity
(95.7% of consistent participants)



58.6% of participants used words consistently and respected all biases



More ad-interim conclusions

- Humans are not perfect composers either...
- But they display different problems from those that challenge neural networks
- Are human biases useful for fast learning?
- Can we get neural networks to display the same biases?

Outline

- Recurrent neural networks
- A compositional challenge for neural networks (and humans)
- **(If time allows) Looking for a compositional neural network in a haystack**

Can a generic RNN learn to behave compositionally?

Adam Liska, Germán Kruszewski and Marco Baroni. Memorize or generalize? Searching for a compositional RNN in a haystack.

AEGAP Workshop 2018

The table lookup domain

t1	t2	t3	t4	t5	...
00 → 10	00 → 11	00 → 00	00 → 11	00 → 10	
01 → 11	01 → 00	01 → 01	01 → 10	01 → 00	
10 → 01	10 → 01	10 → 10	10 → 00	10 → 01	
11 → 00	11 → 10	11 → 11	11 → 01	11 → 11	

t1(00)=10
t3(00)=00

t4(t5(01))=11
t5(t4(01))=01
t2(t2(10))=00

t1(t4(t5(11)))=11
t1(t5(t1(10)))=10

The table lookup domain

t1

00	➡	10
01	➡	11
10	➡	01
11	➡	00

t2

00	➡	11
01	➡	00
10	➡	01
11	➡	10

t3

00	➡	00
01	➡	01
10	➡	10
11	➡	11

t4

00	➡	11
01	➡	10
10	➡	00
11	➡	01

t5

00	➡	10
01	➡	00
10	➡	01
11	➡	11

...

t1(00)=10
t3(00)=00

nothing smart about primitive lookup learning: tables can only be memorized

t4(t5(01))=11
t5(t4(01))=01
t2(t2(10))=00

infinite expressions by finite means

t1(t4(t5(11)))=11
t1(t5(t1(10)))=10

Testing compositional generalization

	t1	t2	t3	t4	t5	...
00 ⇨	10	11	00	11	10	
01 ⇨	11	00	01	10	00	
10 ⇨	01	01	10	00	01	
11 ⇨	00	10	11	01	11	

red = must be generated by RNN

Training phase #1: simple lookups

t1:00.**10.** t4:10.**00.** t301.**01.** ...

Training phase #2: simple and composed lookups

ct1t4:00:**00.** t3:10.**10.** ct5t5:01.**10.** ...

Test phase: composed lookups seen during training, with **novel** inputs:

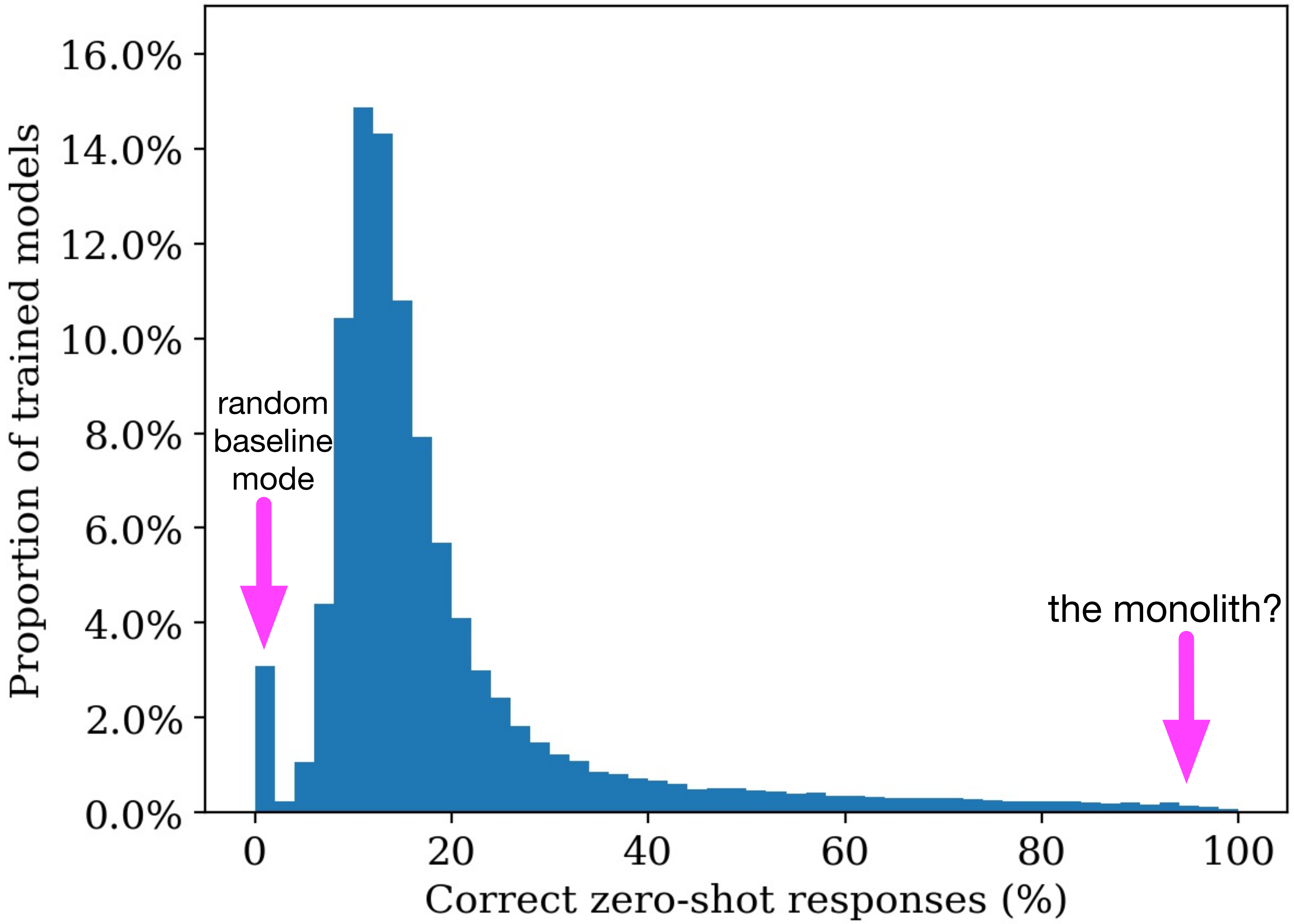
ct1t4:01:**01.** ct5t5:00.**01.** ct3t2:10.**01.**

figure of merit:
0-shot accuracy

Experimental setup

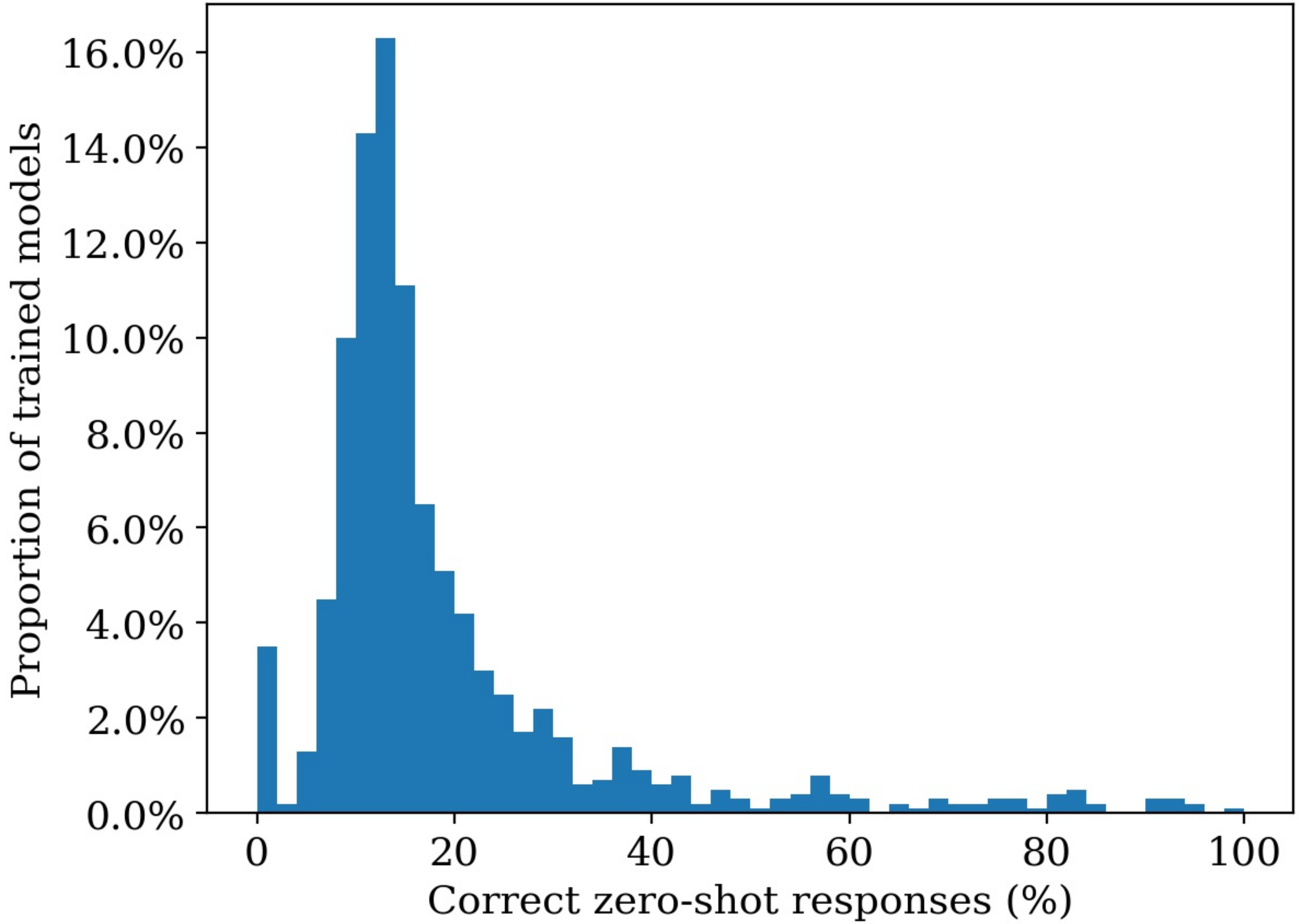
- Recurrent network with two hidden layers
  - Recurrent 60-unit LSTM layer
  - 10-unit sigmoid layer
  - **This architecture *can theoretically* encode a compositional solution**
- Model reads instructions and produces output character-by-character
  - RNN's own output at *t-1* also fed with input at *t*
- Experimenting with 3-bit tables, first-order composition only:
  - 1M examples in training phases #1 and #2
  - 128 inputs left-out for testing (2 per possible first-order table composition)
- Standard training: back-propagate cross-entropy loss and update parameters with stochastic gradient descent (parallel updates from 40 CPUs)
- Experiment repeated 50k times from random initializations
  - From uniform [-0.1, 0.1] range

Looking for a compositional RNN in a haystack



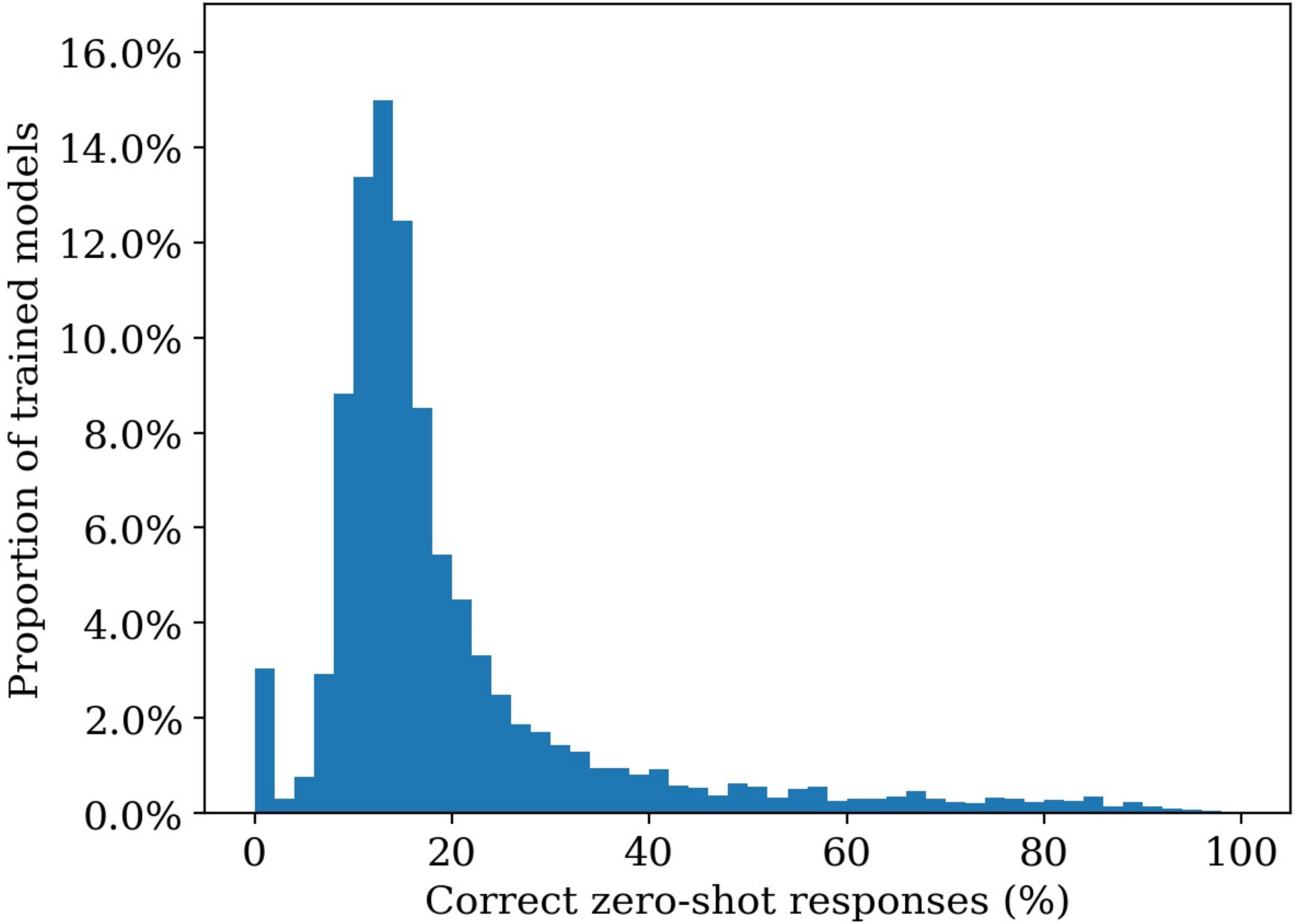
The compositional RNN in a haystack

Same initialization, different runs



The compositional RNN in a haystack

Making the prompts opaque



e.g., composition of t1 and t2 is denoted by ct5t4 instead of ct1t2

no sign of Fregean compositionality!

Conclusion 1

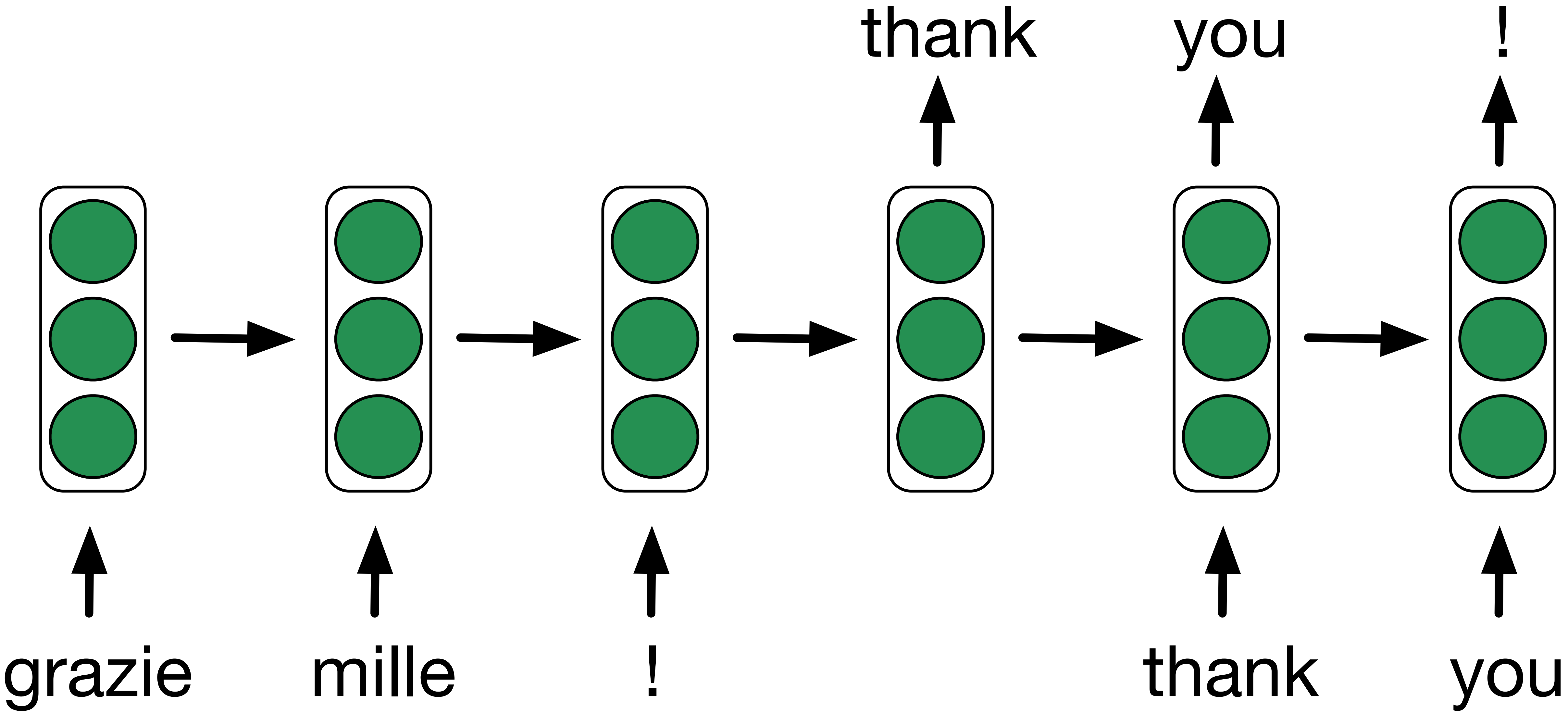
- (Recurrent) neural networks are remarkably powerful and general
  - Agnostic "end-to-end" learners from input-output pairs
- They can generalize to new inputs that are different from those they were trained on...
- ... but their generalization skills do not display **systematic compositionality**
  - Thus, they cannot adapt fast to continuous stream of new inputs in domains such as language, math, and more generally reasoning

Conclusion 2

- We could hard-code compositionality into neural network architectures...
- ... but this might dramatically affect their generality and effectiveness
  - Each new domain will require a new hand-coded set of modules and composition rules
  - Generic (recurrent) neural networks are still the workhorse of successful deep learning applications to language
- General RNN architecture can learn to encode partially compositional solutions
- ... but standard training methods do not easily converge to such solutions

Conclusion 3

- We don't have a full understanding of how compositional reasoning works in humans
- Our preliminary evidence (and work by others) suggests biases in human compositional reasoning
  - What are the biases at work?
  - Are they important to learn to perform compositional reasoning?
  - Should we inoculate them into artificial neural networks?


thank
you
!
grazie
mille
!
thank
you