Tuesday • October 5, 2021

Some final thoughts on Word2Vec and Recurrent Networks

Yale

LING 380/780

Neural Network Models of Linguistic Structure

Other applications of recurrent networks

- Autoregressive generation
- Sequence labeling
- Sequence acceptance/classification
- Sequence to sequence mapping

The Shape of MLPs

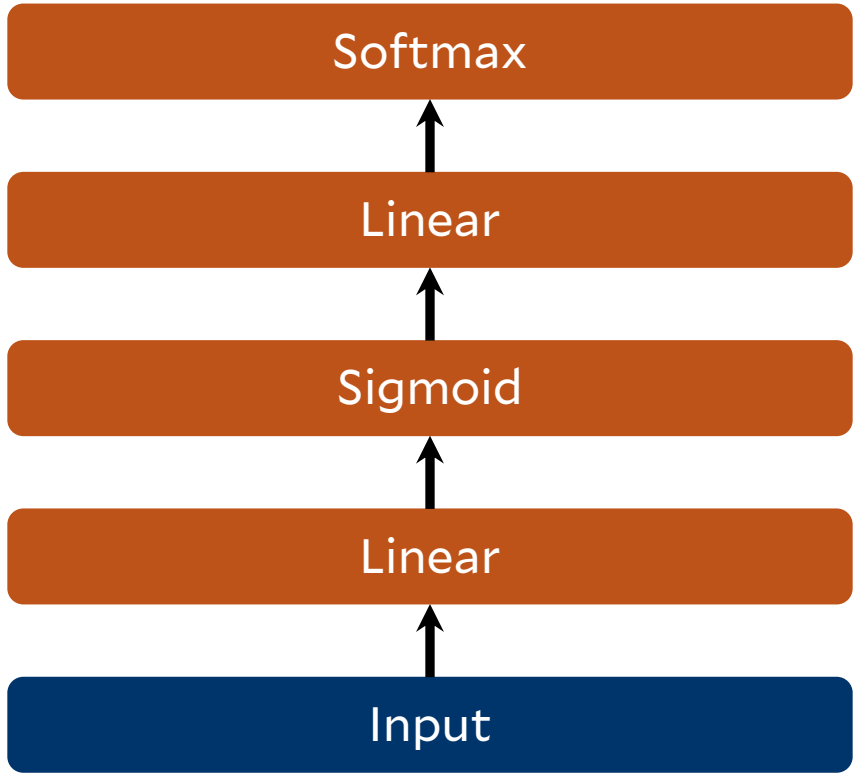
The interfaces to MLPs (and other similar networks) are characterized by the dimensionality of their weight matrices:

$$\boldsymbol{h} = \boldsymbol{\sigma}(\boldsymbol{W}^{(h)}\boldsymbol{x} + \boldsymbol{b}^{(h)})$$

$$\boldsymbol{y} = \mathrm{softmax}\left(\boldsymbol{W}^{(o)}\boldsymbol{h} + \boldsymbol{b}^{(o)}\right)$$

If $\boldsymbol{W}^{(h)}$ is $m \times n$, input is of size n

If $\boldsymbol{W}^{(o)}$is $k \times m$, output is of size k



The Impact of Fixed Dimensionality

- Classification (e.g., NER)

  Brazil 's health minister has tested positive for the coronavirus while in New York for the United Nations General Assembly , where President Jair Bolsonaro spoke on Tuesday .

The Impact of Fixed Dimensionality

- Prediction (Language Modeling): given a context, what is the next word?
  - Selectional restrictions

I walked to the ____

talked

N-grams

- This is analogous to Markovian assumption about the role of context: word depends only on the preceding k words:

$$P(w^{(t)} | w^{(1)} w^{(2)} \dots w^{(t-1)}) = P(w^{(t)} | w^{(t-k)} \dots w^{(t-1)})$$

$$= \frac{P(w^{(t-k)} \dots w^{(t-1)} w^{(t)})}{P(w^{(t-k)} \dots w^{(t-1)})}$$

$$\approx \frac{count(w^{(t-k)} \dots w^{(t-1)} w^{(t)})}{count(w^{(t-k)} \dots w^{(t-1)})}$$

N-grams

- Estimating probabilities from a corpus:

$$P(store|walked\ to\ the) = \frac{count(walked\ to\ the\ store)}{count(walked\ to\ the)}$$

- Some trigram text (trained on *Moby Dick*):

  I would fain kill all his host were now bent and reefed and a spare Bible for the barbs their final heat and wet have they have nevertheless furnished both nations with the standing opulence like fully ripe grapes their wine as I popped out of the mate and his far fiercer curse into the boats at once “Ay ay sir A shoal of Sperm Whales editions all of ye nor can the prisoner reach .

Problems with N-grams

- **Sparsity**: An n-gram didn't occur (or not sufficiently frequently to ensure reliable estimates)?
- **Model size**: Increasing n, increases the number of stored parameters $O(|V|^n)$

A fixed-dimensional language model (Bengio et al., 2002)

output distribution

$$\hat{\boldsymbol{y}} = \text{softmax}(\boldsymbol{U}\boldsymbol{h} + \boldsymbol{b}_2) \in \mathbb{R}^{|V|}$$

hidden layer

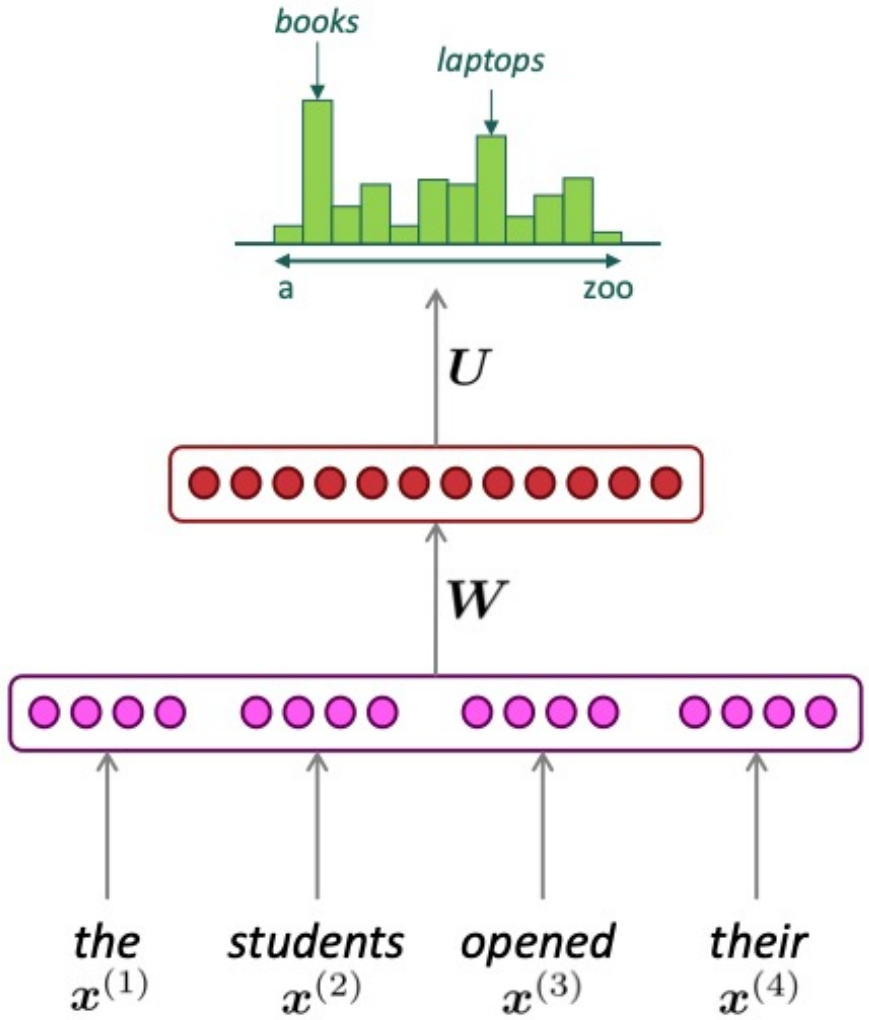
$$\boldsymbol{h} = f(\boldsymbol{W}\boldsymbol{e} + \boldsymbol{b}_1)$$

concatenated word embeddings

$$\boldsymbol{e} = [\boldsymbol{e}^{(1)}; \boldsymbol{e}^{(2)}; \boldsymbol{e}^{(3)}; \boldsymbol{e}^{(4)}]$$

words / one-hot vectors

$$\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}, \boldsymbol{x}^{(4)}$$



- Sparsity?
- Model size?

The Impact of Fixed Dimensionality

- Prediction (Language Modeling)
  - Agreement

    The book(s) _____
    is
    are

The Impact of Fixed Dimensionality

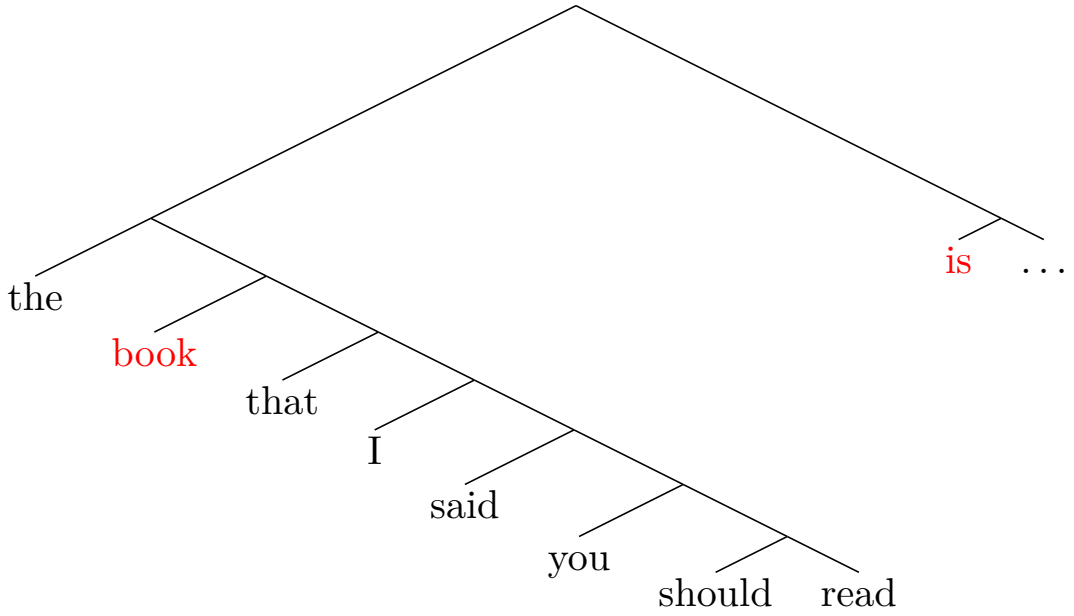
- Prediction (Language Modeling)
  - Agreement

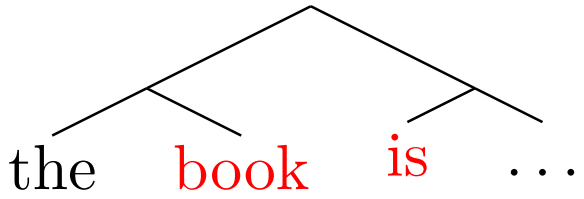
Unbounded *linear* dependency

The book(s) that I said you should read _____

is

are





The Impact of Fixed Dimensionality

- Prediction (Language Modeling)
  - Displacement

you think I should devour _____

cookies

[EOS]

I know what you think I should devour _____

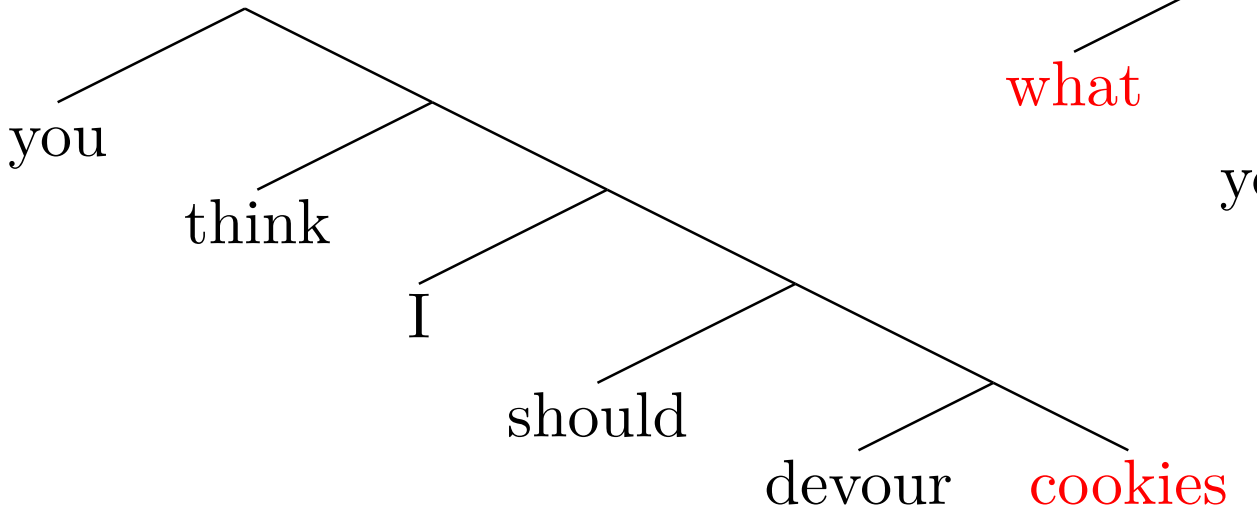
cookies

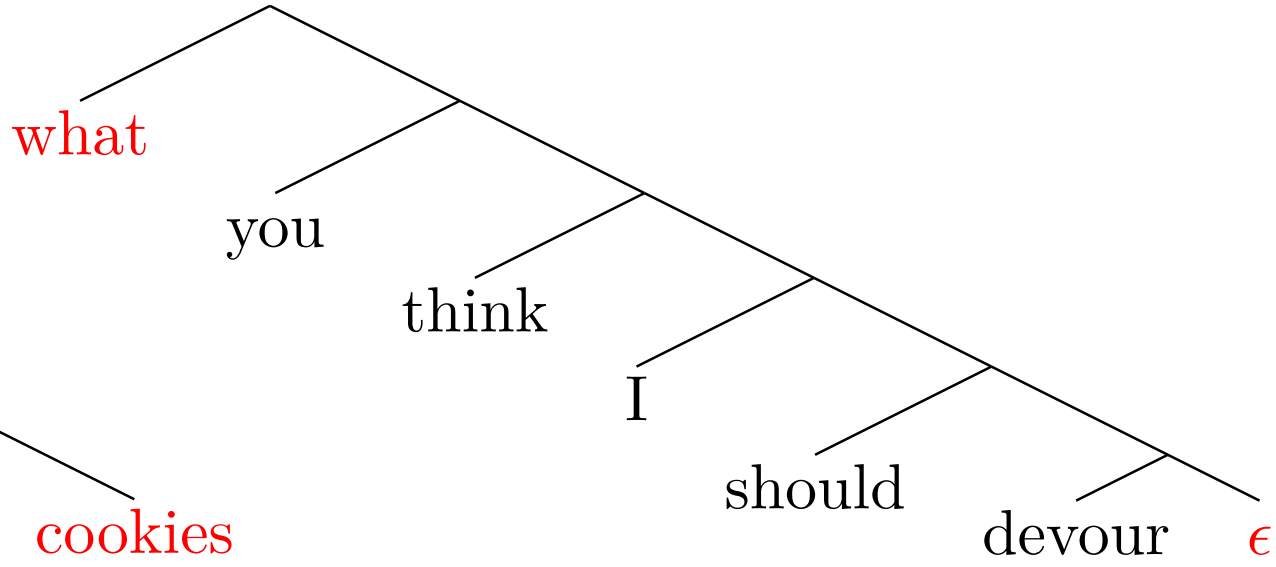
[EOS]

The Impact of Fixed Dimensionality

- Prediction (Language Modeling)
  - Displacement

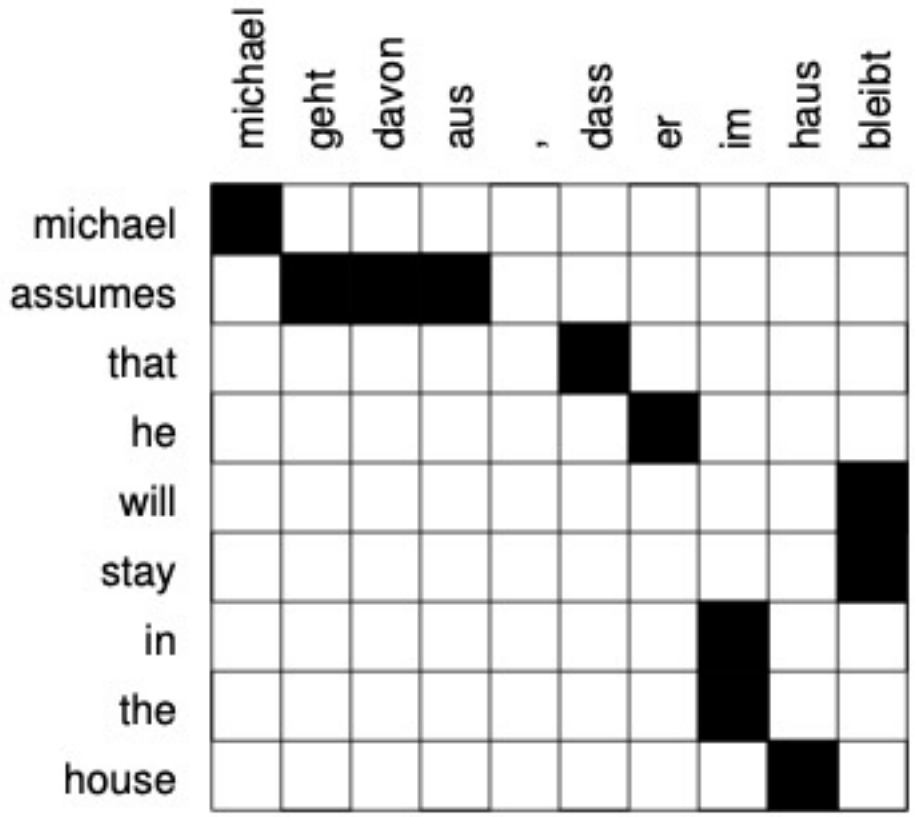
Unbounded *structural* dependency





The Impact of Fixed Dimensionality

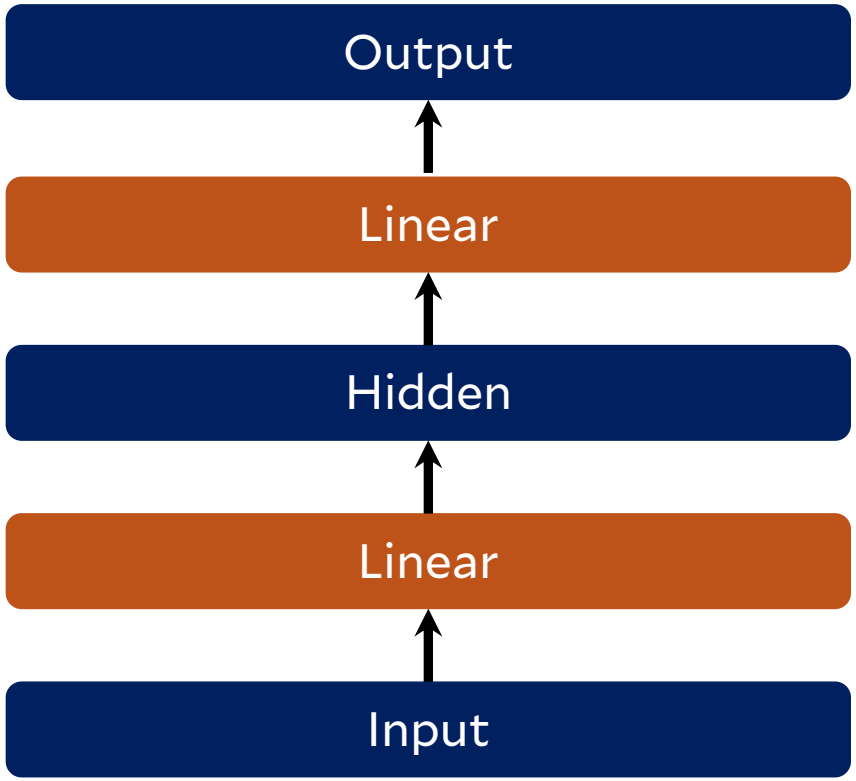
- Translation: many-to-many mappings

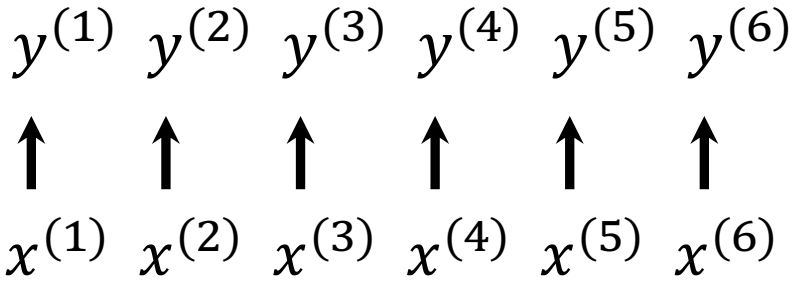


Requires unbounded inputs and outputs

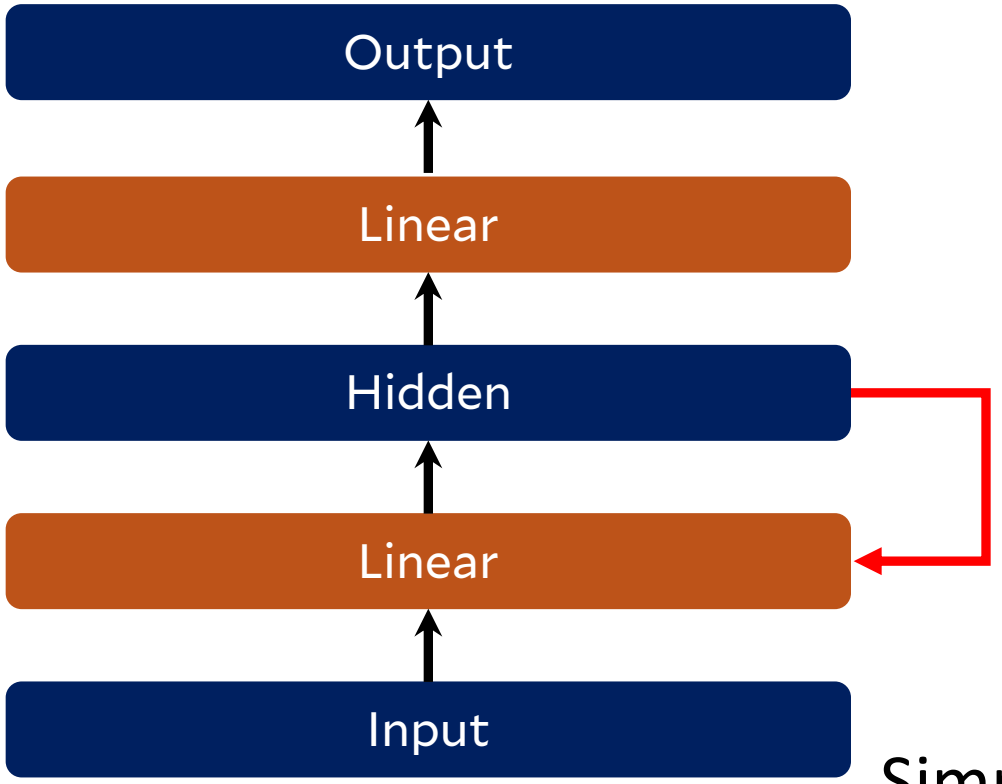
A solution: unboundedness in time

$$\boldsymbol{h} = \boldsymbol{\sigma}(\boldsymbol{W}^{(h)}\boldsymbol{x} + \boldsymbol{b}^{(h)})$$
$$\boldsymbol{y} = \mathrm{softmax}\left(\boldsymbol{W}^{(o)}\boldsymbol{h} + \boldsymbol{b}^{(o)}\right)$$





Unboundedness in time

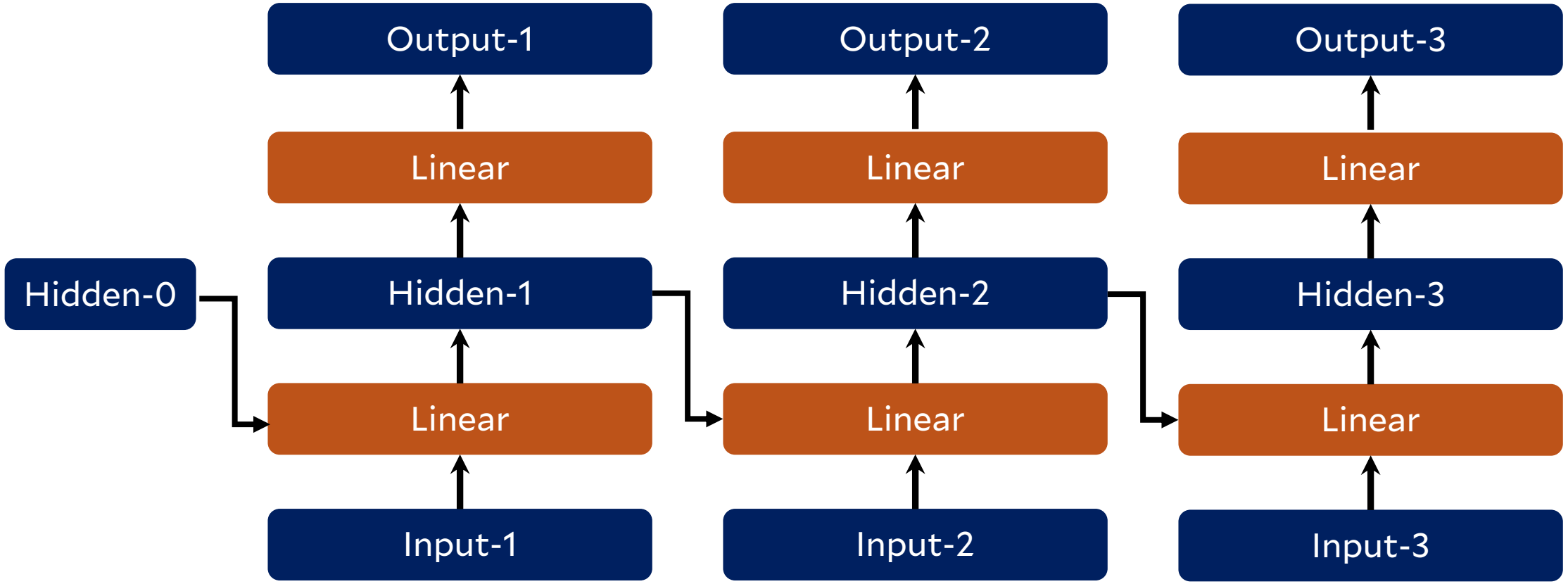


$$\boldsymbol{h} = \boldsymbol{\sigma}(\boldsymbol{W}^{(h)}\boldsymbol{x} + \boldsymbol{b}^{(h)})$$
$$\boldsymbol{y} = \mathrm{softmax}\left(\boldsymbol{W}^{(o)}\boldsymbol{h} + \boldsymbol{b}^{(o)}\right)$$

$$\boldsymbol{h}^{(t)} = \boldsymbol{\sigma}(\boldsymbol{W}^{(h)}\boldsymbol{x}^{(t)} + \boldsymbol{U}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}^{(h)})$$
$$\boldsymbol{y}^{(t)} = \mathrm{softmax}\left(\boldsymbol{W}^{(o)}\boldsymbol{h}^{(t)} + \boldsymbol{b}^{(o)}\right)$$

Simple Recurrent Network (SRN/RNN)
(Elman 1990)

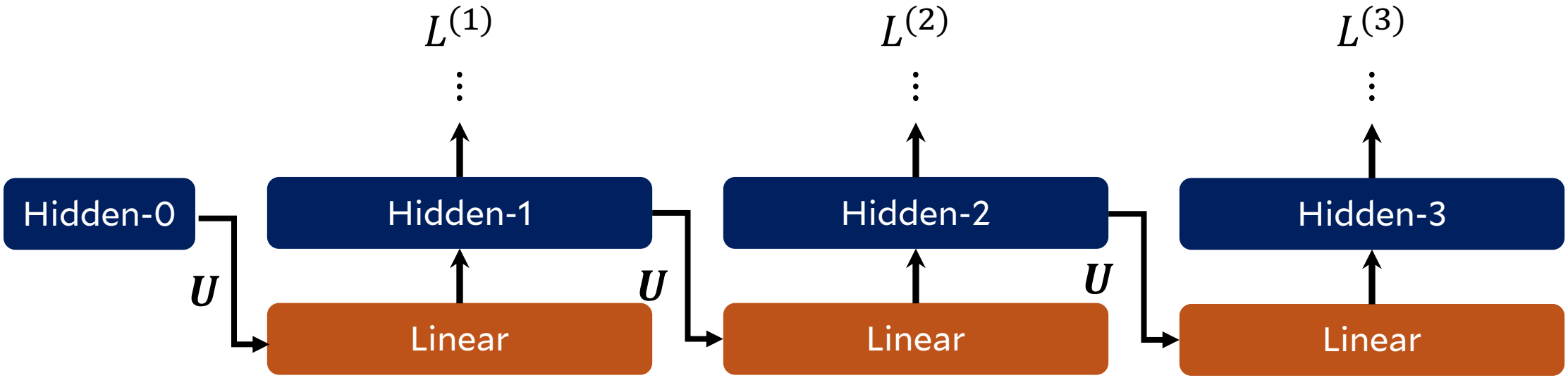
Unrolling the RNN: time becomes space



Training an RNN

- Train to predict the next word in a text (“self supervised”)
- Use SGD over computation graph:
  - Run forward computation for a sentence (or batch of sentences) and compute average loss
  - Compute gradients of loss with respect to model parameters.
  - But how do we do this when the same parameter shows up in multiple places?

Training an RNN



What is $\frac{\partial L^{(3)}}{\partial \boldsymbol{U}}$?

Answer:

$$\frac{\partial L^{(3)}}{\partial \boldsymbol{U}} = \sum_{i=1}^{3} \frac{\partial L^{(3)}}{\partial \boldsymbol{U}}\bigg|_{(i)}$$

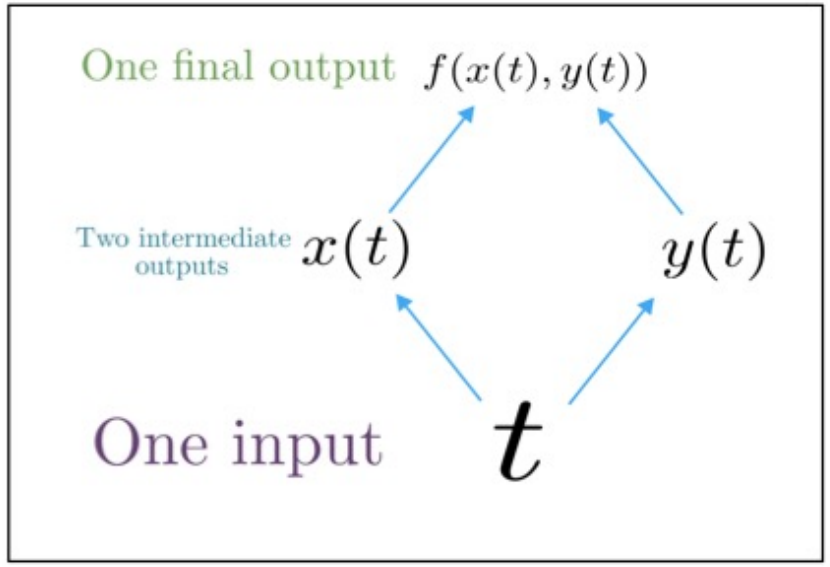
In other words:

The gradient with respect to a repeated weight is the sum of the gradient with respect to each of its occurrences in the computation graph

Multivariable chain rule

- Given a multivariable function $f(x, y)$ and two single variable functions $x(t)$ and $y(t)$:

$$\frac{d}{dt} f\big(x(t), y(t)\big) = \frac{\partial f}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial t}$$



Multivariable chain rule

$$\frac{d}{dt} f\big(x(t), y(t)\big) = \frac{\partial f}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial t}$$

$$\boldsymbol{h}^{(t)} = \sigma(\boldsymbol{W}^{(h)}\boldsymbol{x}^{(t)} + \boldsymbol{U}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}^{(h)})$$
$$\boldsymbol{y}^{(t)} = \text{softmax}\big(\boldsymbol{W}^{(o)}\boldsymbol{h}^{(t)} + \boldsymbol{b}^{(o)}\big)$$
$$\boldsymbol{L}^{(t)} = -\log(\boldsymbol{y}^{(t)})$$

$$\frac{\partial L^{(3)}}{\partial \boldsymbol{U}} = \frac{\partial L^{(3)}}{\partial \boldsymbol{y}^{(3)}}\frac{\partial \boldsymbol{y}^{(3)}}{\partial \boldsymbol{h}^{(3)}}\frac{\partial \boldsymbol{h}^{(3)}}{\partial \boldsymbol{U}}$$

$\boldsymbol{h}^{(3)}$ depends on $\boldsymbol{U}$ and on $\boldsymbol{h}^{(2)}$ (which depends on $\boldsymbol{U}$)

$$\frac{\partial h^{(3)}}{\partial \boldsymbol{U}} = \frac{\partial \sigma(\boldsymbol{W}^{(h)}\boldsymbol{x}^{(3)} + \boldsymbol{U}\boldsymbol{h}^{(2)} + \boldsymbol{b}^{(h)})}{\partial \boldsymbol{U}}\frac{\partial \boldsymbol{U}}{\partial \boldsymbol{U}} + \frac{\partial \sigma(\boldsymbol{W}^{(h)}\boldsymbol{x}^{(3)} + \boldsymbol{U}\boldsymbol{h}^{(2)} + \boldsymbol{b}^{(h)})}{\partial \boldsymbol{h}^{(2)}}\frac{\partial \boldsymbol{h}^{(2)}}{\partial \boldsymbol{U}}$$

Multivariable chain rule

$$\frac{\partial h^{(3)}}{\partial \boldsymbol{U}} = \frac{\partial \sigma(\boldsymbol{W}^{(h)}\boldsymbol{x}^{(3)} + \boldsymbol{U}\boldsymbol{h}^{(2)} + \boldsymbol{b}^{(h)})}{\partial \boldsymbol{U}}\frac{\partial \boldsymbol{U}}{\partial \boldsymbol{U}} + \frac{\partial \sigma(\boldsymbol{W}^{(h)}\boldsymbol{x}^{(3)} + \boldsymbol{U}\boldsymbol{h}^{(2)} + \boldsymbol{b}^{(h)})}{\partial \boldsymbol{h}^{(2)}}\frac{\partial \boldsymbol{h}^{(2)}}{\partial \boldsymbol{U}}$$

$$= \frac{\partial \sigma(\boldsymbol{W}^{(h)}\boldsymbol{x}^{(3)}+\boldsymbol{U}\boldsymbol{h}^{(2)}+\boldsymbol{b}^{(h)})}{\partial \boldsymbol{U}} +$$

$$\frac{\partial \sigma(\boldsymbol{W}^{(h)}\boldsymbol{x}^{(3)}+\boldsymbol{U}\boldsymbol{h}^{(2)}+\boldsymbol{b}^{(h)})}{\partial \boldsymbol{h}^{(2)}}\left(\frac{\partial \sigma(\boldsymbol{W}^{(h)}\boldsymbol{x}^{(2)}+\boldsymbol{U}\boldsymbol{h}^{(1)}+\boldsymbol{b}^{(h)})}{\partial \boldsymbol{U}} + \frac{\partial \sigma(\boldsymbol{W}^{(h)}\boldsymbol{x}^{(2)}+\boldsymbol{U}\boldsymbol{h}^{(2)}+\boldsymbol{b}^{(h)})}{\partial \boldsymbol{h}^{(1)}}\frac{\partial \boldsymbol{h}^{(1)}}{\partial \boldsymbol{U}}\right)$$

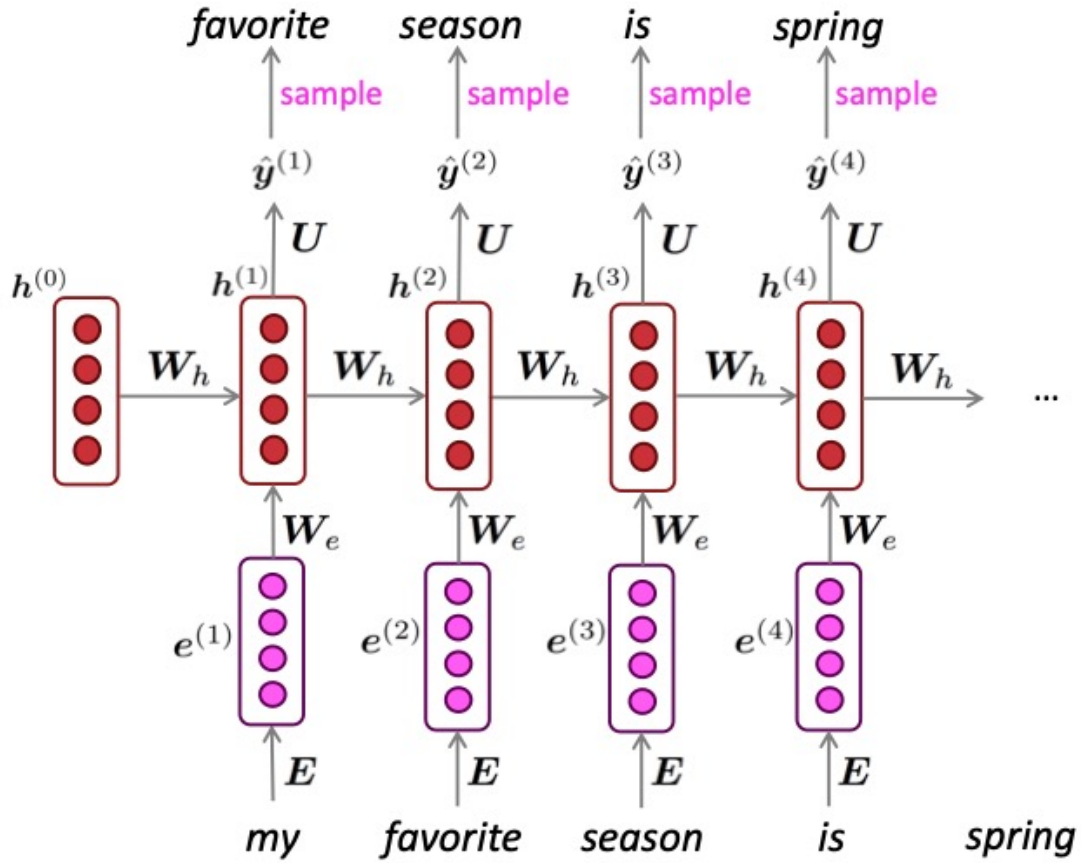
Etc.

Training an RNN

- We can compute this by starting at time t, and computing gradients on successively preceding time steps, summing gradients as you proceed.
- This algorithm is called Backprop Through Time (BPTT):
- To limit the explosion of terms, we often truncate the number of steps through which gradient propagate

RNN Language models

- RNN Language models can be used as language generators:



RNN language models

- Trained on Barack Obama speeches:

  The United States will step up to the cost of a new challenges of the American people that will share the fact that we created the problem. They were attacked and so that they have to say that all the task of the final days of war that I will not be able to get this done. The promise of the men and women who were still going to take out the fact that the American people have fought to make sure that they have to be able to protect our part. It was a chance to stand together to completely look for the commitment to borrow from the American people.

https://medium.com/@samim/obama-rnn-machine-generated-political-speeches-c8abd18a2ea0

RNN language models

- Trained on recipes:

Title: CARAMEL CORN GARLIC BEEF
Categories: Soups, Desserts
Yield: 10 Servings

2 tb Parmesan cheese, ground
1/4 ts Ground cloves
-- diced
1 ts Cayenne pepper

Cook it with the batter. Set aside to cool. Remove the peanut oil in a small saucepan and pour into the margarine until they are soft. Stir in a a mixer (dough). Add the chestnuts, beaten egg whites, oil, and salt and brown sugar and sugar; stir onto the boqtly brown it.

The recipe from an oiled by fried and can. Beans, by Judil Cookbook, Source: Pintore, October, by Chocolates, Breammons of Jozen, Empt.com

https://gist.github.com/nylki/1efbaa36635956d35bcc

RNN Language models

- We measure the quality of a language model $\boldsymbol{P_{LM}}$ with **perplexity**

$$\boldsymbol{perplexity}(\boldsymbol{C}) = \sqrt[N]{\boldsymbol{P_{LM}}(\boldsymbol{C})}$$

$$\boldsymbol{P_{LM}}(\boldsymbol{C}) = \prod_{\boldsymbol{i=1}}^{\boldsymbol{N}} \boldsymbol{P_{LM}}(\boldsymbol{w^{(i)}})$$

$$= \exp(\log(\prod_{\boldsymbol{i=1}}^{\boldsymbol{N}} \boldsymbol{P_{LM}}(\boldsymbol{w^{(i)}})^{-\frac{1}{N}}))$$

$$= \exp(-\frac{1}{N}\log(\prod_{\boldsymbol{i=1}}^{\boldsymbol{N}} \boldsymbol{P_{LM}}(\boldsymbol{w^{(i)}})))$$

$$= \exp(\frac{1}{N}\sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} -\log(\boldsymbol{P_{LM}}(\boldsymbol{w^{(i)}})))$$

w is word
P(w_i) is the probability assigned to w_i by the language model.

Exponential of the average cross-entropy loss

RNN Language Models

- Perplexity on the Wall Street Journal corpus (Mikolov 2010):
  - N-gram: 80
  - SRN: 59

RNN Language Models

- Current RNN models do even better (gigaword corpus)

Model	Perplexity
Interpolated Kneser-Ney 5-gram (Chelba et al., 2013)	67.6
RNN-1024 + MaxEnt 9-gram (Chelba et al., 2013)	51.3
RNN-2048 + BlackOut sampling (Ji et al., 2015)	68.3
Sparse Non-negative Matrix factorization (Shazeer et al., 2015)	52.9
LSTM-2048 (Jozefowicz et al., 2016)	43.7
2-layer LSTM-8192 (Jozefowicz et al., 2016)	30
Ours small (LSTM-2048)	43.9
Ours large (2-layer LSTM-2048)	39.8

https://research.fb.com/building-an-efficient-neural-language-model-over-a-billion-words/

What are they learning?

Distributed representations, simple recurrent networks, and grammatical structure
https://link.springer.com/article/10.1007/BF00114844

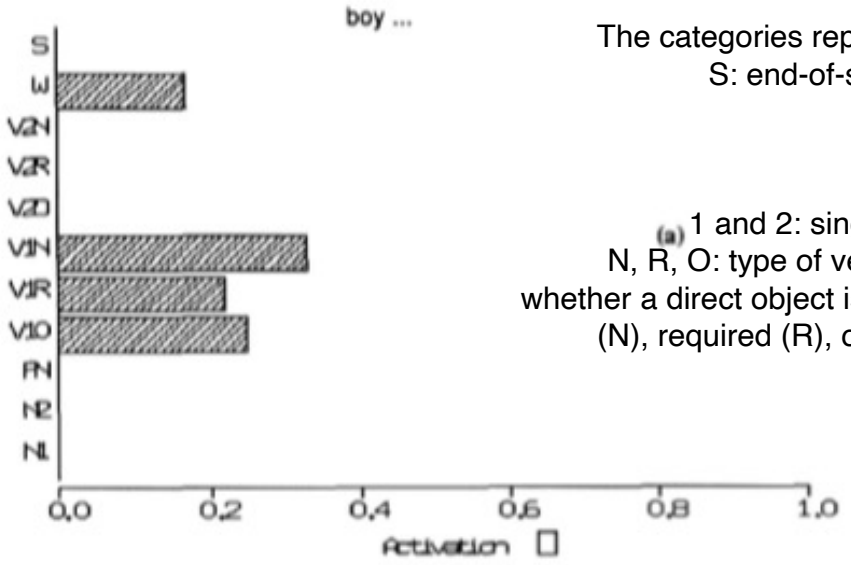
- Elman (1991) – synthetic language
  - Agreement:
    - John feeds dogs.
    - Boys see dogs.
  - Subcategorization:
    - Girls feed dogs. (obligatorily transitive)
    - Girls live. (obligatorily intransitive)
    - Girls see (optionally transitive)
  - Relative Clauses:
    - Dogs who chase cats see dogs.
    - Dogs who Mary chases see dogs.

Learning agreement

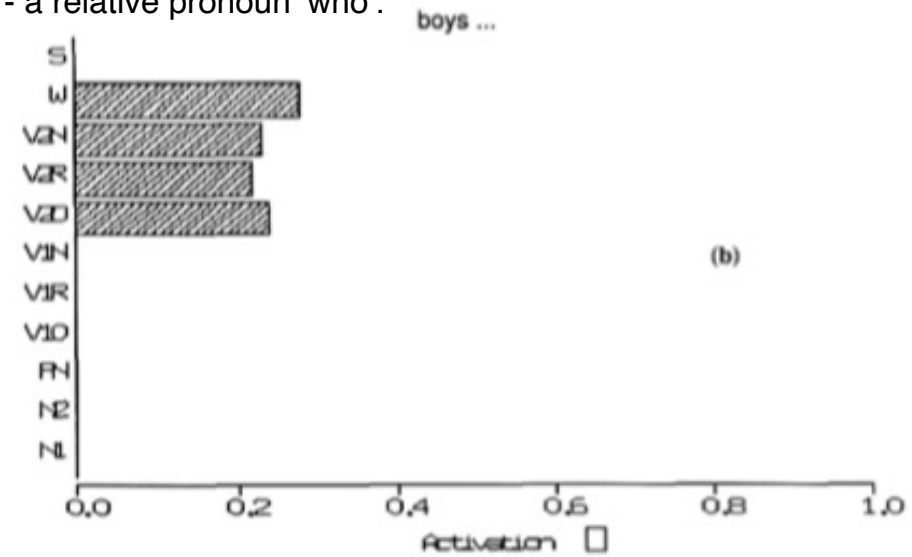
the next word following a word "boy" (singular) and "boys" (plural) predicted by a language model are visualized as distribution of activations for words grouped by category.

- if context is word ‘boy’, predicted word is
  - a singular verb V1N, V1R, V1D. words in all three singular verb categories are activated, since it has no basis for predicting the type of verb.
  - a relative pronoun ‘who’.

- if context is word ‘boys’, predicted word is
  - a plural verb V2N, V2R, V2D
  - a relative pronoun ‘who’.

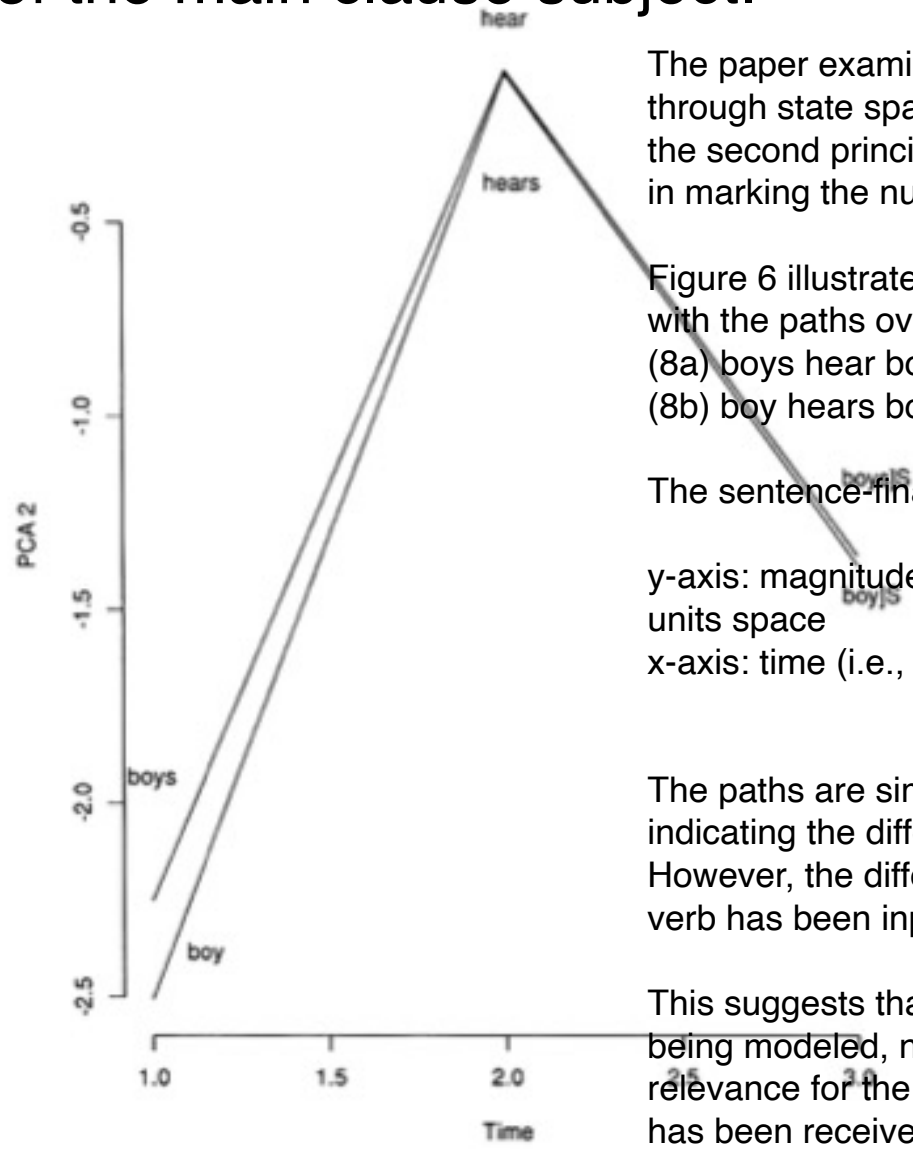


The categories represented are:
S: end-of-sentence (". ")
W: who
N: nouns
V: verbs
1 and 2: singular or plural
N, R, O: type of verb, indicating whether a direct object is not possible (N), required (R), or optional (O)



Text

PCA2 mark the number of the main clause subject.



The paper examined the trajectories of the language model through state space along various dimensions and found that the second principal component PCA2 played an important role in marking the number of the main clause subject.

Figure 6 illustrates the trajectories for sentences (8a) and (8b), with the paths overlaid to highlight their differences.
(8a) boys hear boys.
(8b) boy hears boys.

The sentence-final word is marked with a "]S".

y-axis: magnitude of the second principal component of hidden units space
x-axis: time (i.e., order of the word in the sentence)

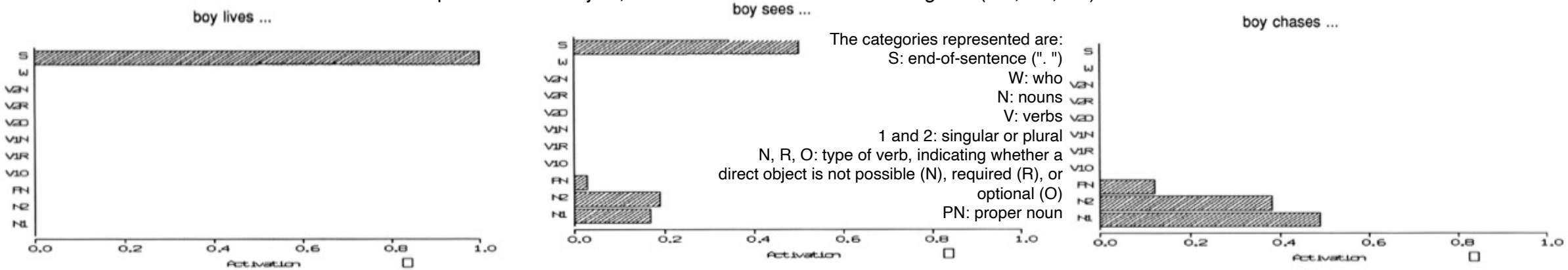
The paths are similar and diverge only during the first word, indicating the difference in the number of the initial noun. However, the difference is slight and is eliminated after the main verb has been input.

This suggests that, for these two sentences and for the grammar being modeled, number information does not have any relevance for the language modeling task once the main verb has been received.

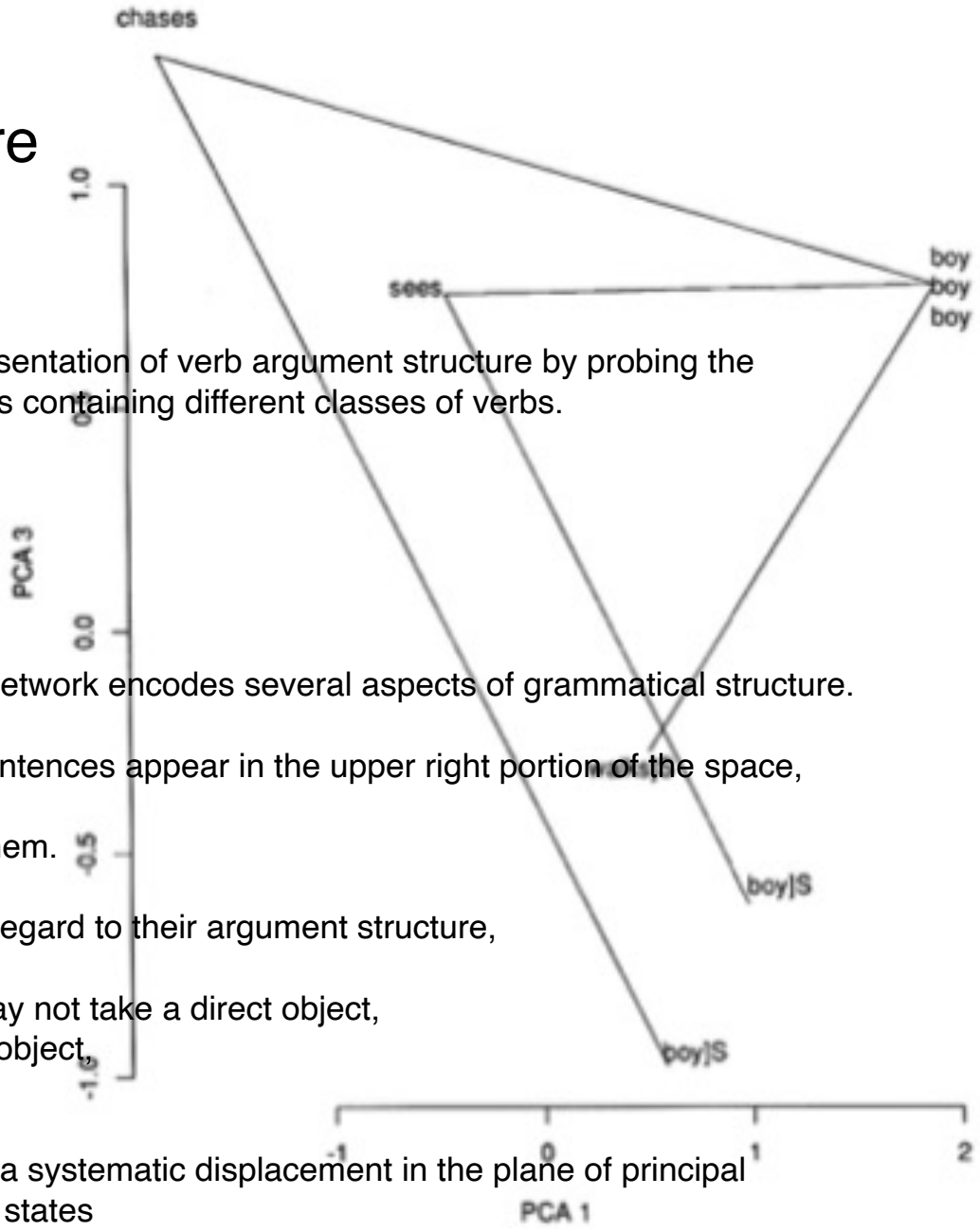
Learning subcategorization

Figure 3 shows predicted next word by network following an initial noun ‘boy’ and then a verb from each of the three different verb types (the first precludes a direct object, the second optional permits a direct object, and the third requires a direct object)

- When the verb is ‘lives’, next word is only ‘.’ end of sentence, which is in fact the only successor permitted by the grammar in this context.
- The verb ‘sees’, next word is either ". ", or optionally a direct object (which may be a singular N1 or plural noun N2, or proper noun PN).
- the verb ‘chases’ requires a direct object, next word will be a noun following this (PN, N1, N2)



verb argument structure



The paper examined the representation of verb argument structure by probing the language model with sentences containing different classes of verbs.

(9a) boy walks.
(9b) boy sees boy.
(9c) boy chases boy.

The figure illustrates how the network encodes several aspects of grammatical structure.

- Subject nouns for all three sentences appear in the upper right portion of the space,

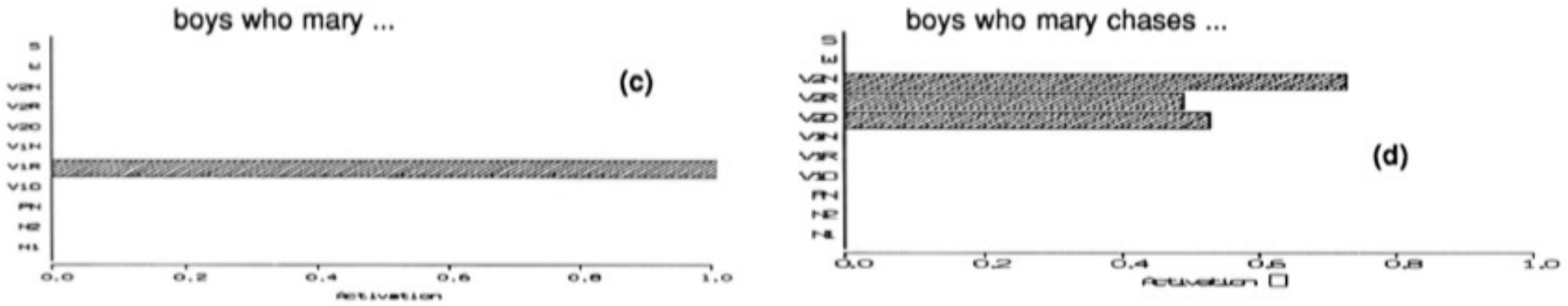
- object nouns appear below them.

- Verbs are differentiated with regard to their argument structure,

  (9a) containing a verb that may not take a direct object,
  (9b) taking an optional direct object,
  (9c) requiring a direct object.

This distinction is reflected in a systematic displacement in the plane of principal components 1 and 3 of hidden states

Learning relative clauses



The paper demonstrates a subtle point in (4c), where the appearance of "boys" followed by a relative clause ‘who Mary’ containing a different subject 主语 (Mary) primes the network to expect that the verb (chases) which follows must be of the class that requires a direct object (V1R), because a direct object 宾语 filler (boys) has already appeared.
In other words, the network responds correctly to the presence of a filler not only by knowing where to expect a gap, but also by learning that when this filler corresponds to the object position in the relative clause, a verb is required that has the appropriate argument structure.
This shows how the language model is able to use information about the structure and syntax of the sentence to make more accurate predictions about the likely next word in a sequence.

Relative clauses

Figure 9c
(10a) boy chases boy.
(10c) boy who chases boy chases boy.

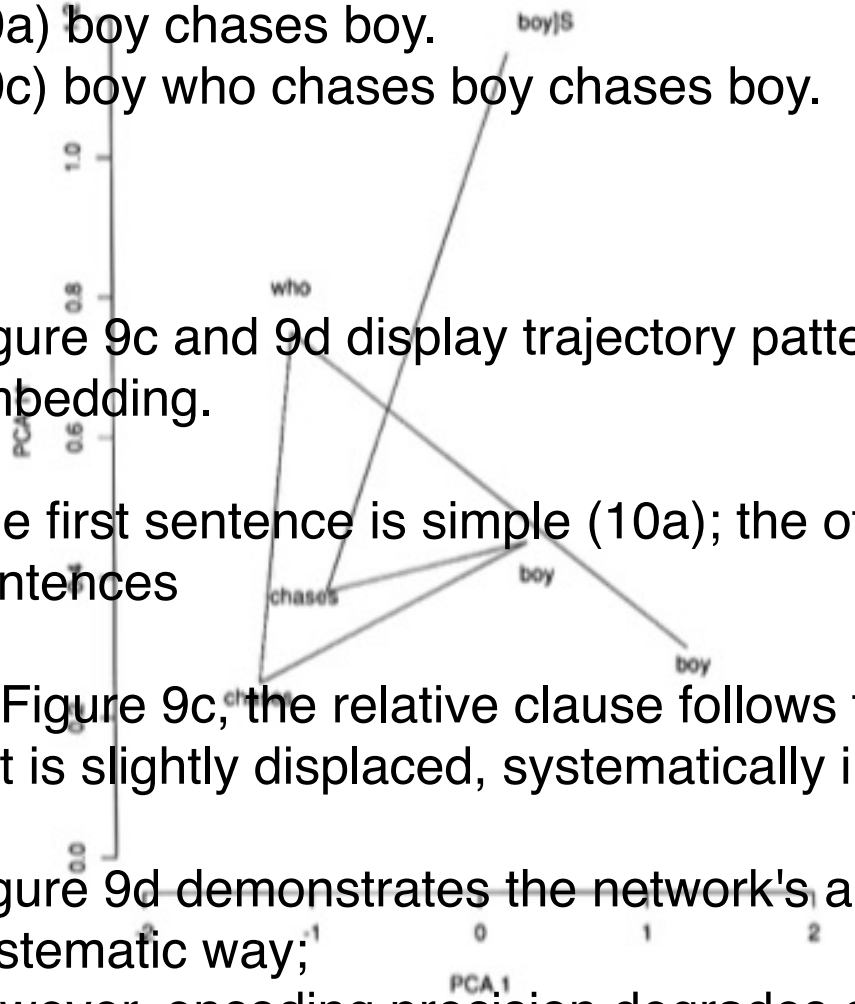


Figure 9d
(10b) boy chases boy who chases boy.
(10d) boy chases boy who chases boy who chases boy

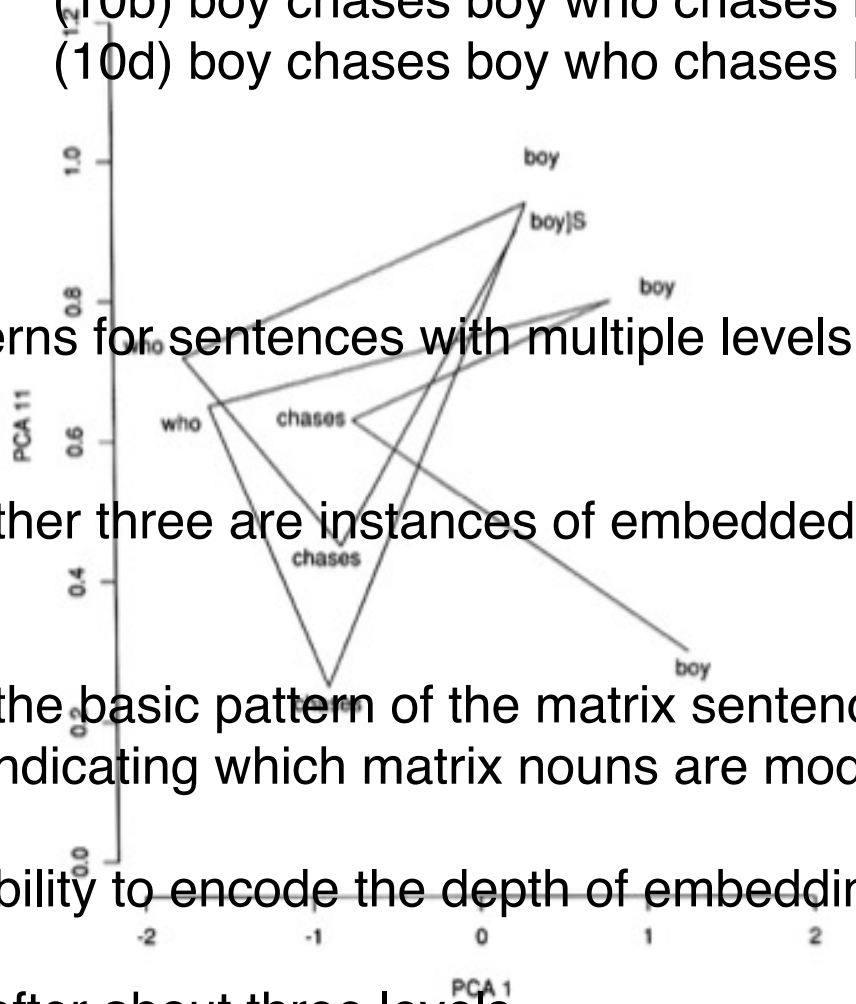


Figure 9c and 9d display trajectory patterns for sentences with multiple levels of embedding.

The first sentence is simple (10a); the other three are instances of embedded sentences

In Figure 9c, the relative clause follows the basic pattern of the matrix sentence 主句, but is slightly displaced, systematically indicating which matrix nouns are modified.

Figure 9d demonstrates the network's ability to encode the depth of embedding in a systematic way;
however, encoding precision degrades after about three levels.
The network's nonlinear nature allows it to remain sensitive to context at the level of internal representation, despite minimizing context in terms of behavior.

Formal languages

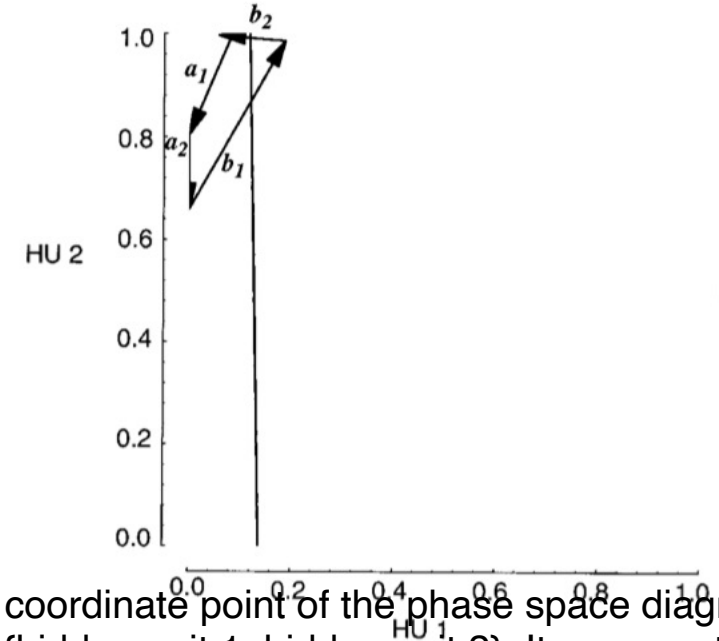
paper "A Recurrent Neural Network that Learns to Count,"

task: input-output mapping.

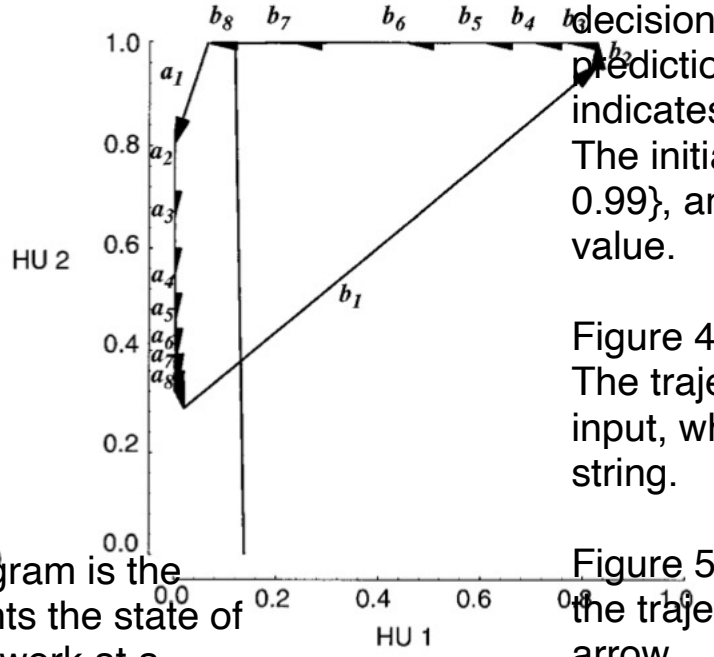
RNN is trained to predict the next symbol ‘a’ or ‘b’ in a sequence.

input sequence: a^n b^n strings. a simple Dyck context-free language (DCFL) with two symbols, {a, b}

- Rodriguez, Wiles and Elman (1999): $a^n b^n$



The coordinate point of the phase space diagram is the pair {hidden unit 1, hidden unit 2}. It represents the state of the two hidden units in a recurrent neural network at a specific time step.



Each arrow represents one time step (i.e., one input value).
The nearly vertical line represents the 0.5 threshold decision boundary, where the left side indicates an "a" prediction at the output nodes and the right side indicates a "b" output prediction.
The initial starting point for the first "a" is about {0.05, 0.99}, and the final "b" input ends at about the same value.

Figure 4 shows the trajectories for Network 1 for n=2. The trajectory crosses the dividing line on the last "b" input, which equates to predicting the start of the next string.

Figure 5 shows the trajectories for Network 1 for n=8. the trajectory crosses the dividing line only on the last arrow

Formal languages

The trajectories oscillate around the fixed points but still manage to cross the dividing line on the last bin put, meaning that the network makes the correct prediction at the end of the sequence.

- Rodriguez, Wiles and Elman (1999): $a^n b^n$