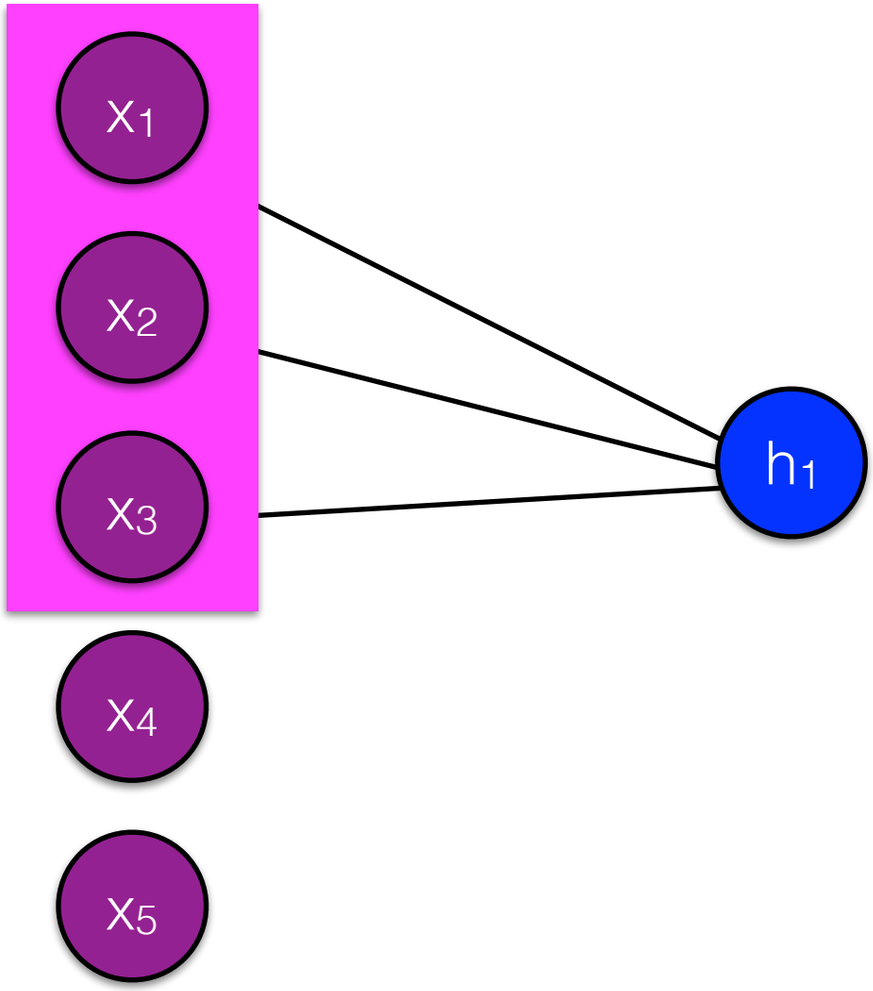
Natural Language Processing

Mehmet Can Yavuz, PhD
Adapted from Info 256 - David Bamman, UC Berkeley

CNNs



	x	W
x_1 / W_1	0.4	3.1
	0.8	-2.7
	1.2	1.4
	-1.3	-0.7
	0.4	-1.4
x_2 / W_2	0.2	9.2
	-5.3	-3.1
	-1.2	-2.7
	5.3	1.4
	0.4	0.1
x_3 / W_3	2.6	0.3
	2.7	-0.4
	-3.2	-2.4
	6.2	-4.7
	1.9	5.7

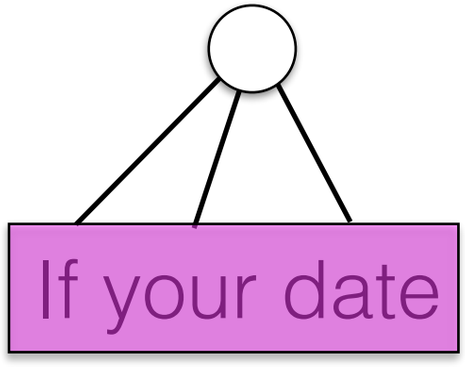
For dense input vectors (e.g., embeddings), full dot product

$$h_1 = \sigma(x_1 W_1 + x_2 W_2 + x_3 W_3)$$

CNNs

- CNNs are limited to creating features scoped over a pre-defined width.

> "*Valentine's Day* is being marketed as a Date Movie. I think it's more of a First-Date Movie. If your date likes it, do not date that person again. And if you like it, there may not be a second date."
>
> Roger Ebert, Valentine's Day



likes it, do not date that person again

if your date
your date likes
date like it
like it ,
it , do

, do not
do not date
not date that
date that person
that person again

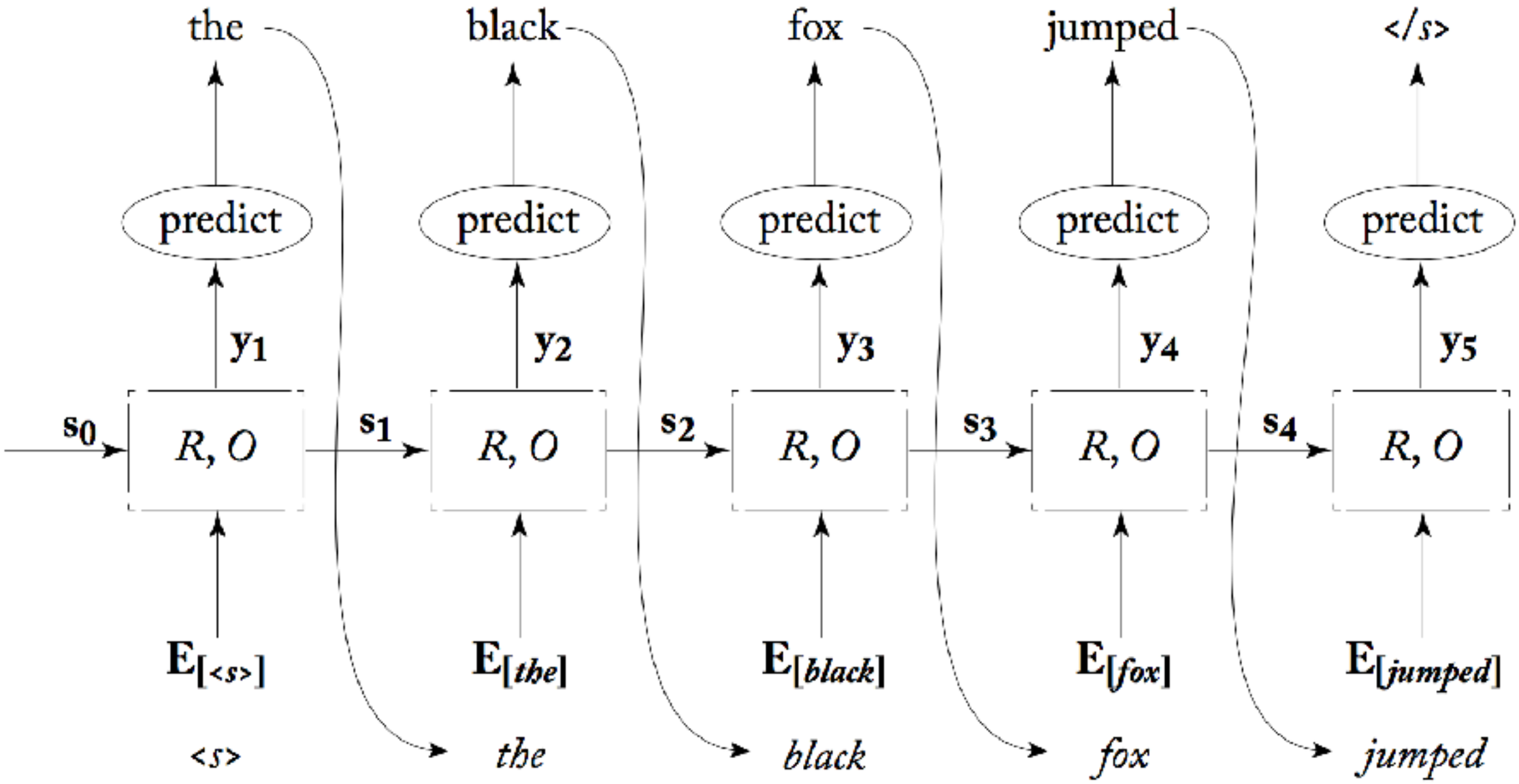
Recurrent neural network

- RNN allow arbitarily-sized conditioning contexts; condition on the entire sequence history.

Recurrent neural network

- Often used for sequential prediction tasks:
    - Language models—predicting the next symbol (word, character) in a sequence

Generation



Goldberg 2017

Character LM

```
/*
 * Increment the size file of the new incorrect UI_FILTER group information
 * of the size generatively.
 */
static int indicate_policy(void)
{
  int error;
  if (fd == MARN_EPT) {
    /*
     * The kernel blank will coeld it to userspace.
     */
    if (ss->segment < mem_total)
      unblock_graph_and_set_blocked();
    else
      ret = 1;
    goto bail;
  }
  segaddr = in_SB(in.addr);
  selector = seg / 16;
  setup_works = true;
```

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Character LM

```
\begin{proof}
We may assume that $\mathcal{I}$ is an abelian sheaf on $\mathcal{C}$.
\item Given a morphism $\Delta : \mathcal{F} \to \mathcal{I}$
is an injective and let $\mathfrak q$ be an abelian sheaf on $X$.
Let $\mathcal{F}$ be a fibered complex. Let $\mathcal{F}$ be a category.
\begin{enumerate}
\item \hyperref[setain-construction-phantom]{Lemma}
\label{lemma-characterize-quasi-finite}
Let $\mathcal{F}$ be an abelian quasi-coherent sheaf on $\mathcal{C}$.
Let $\mathcal{F}$ be a coherent $\mathcal{O}_X$-module. Then
$\mathcal{F}$ is an abelian catenary over $\mathcal{C}$.
\item The following are equivalent
\begin{enumerate}
\item $\mathcal{F}$ is an $\mathcal{O}_X$-module.
\end{lemma}
```

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

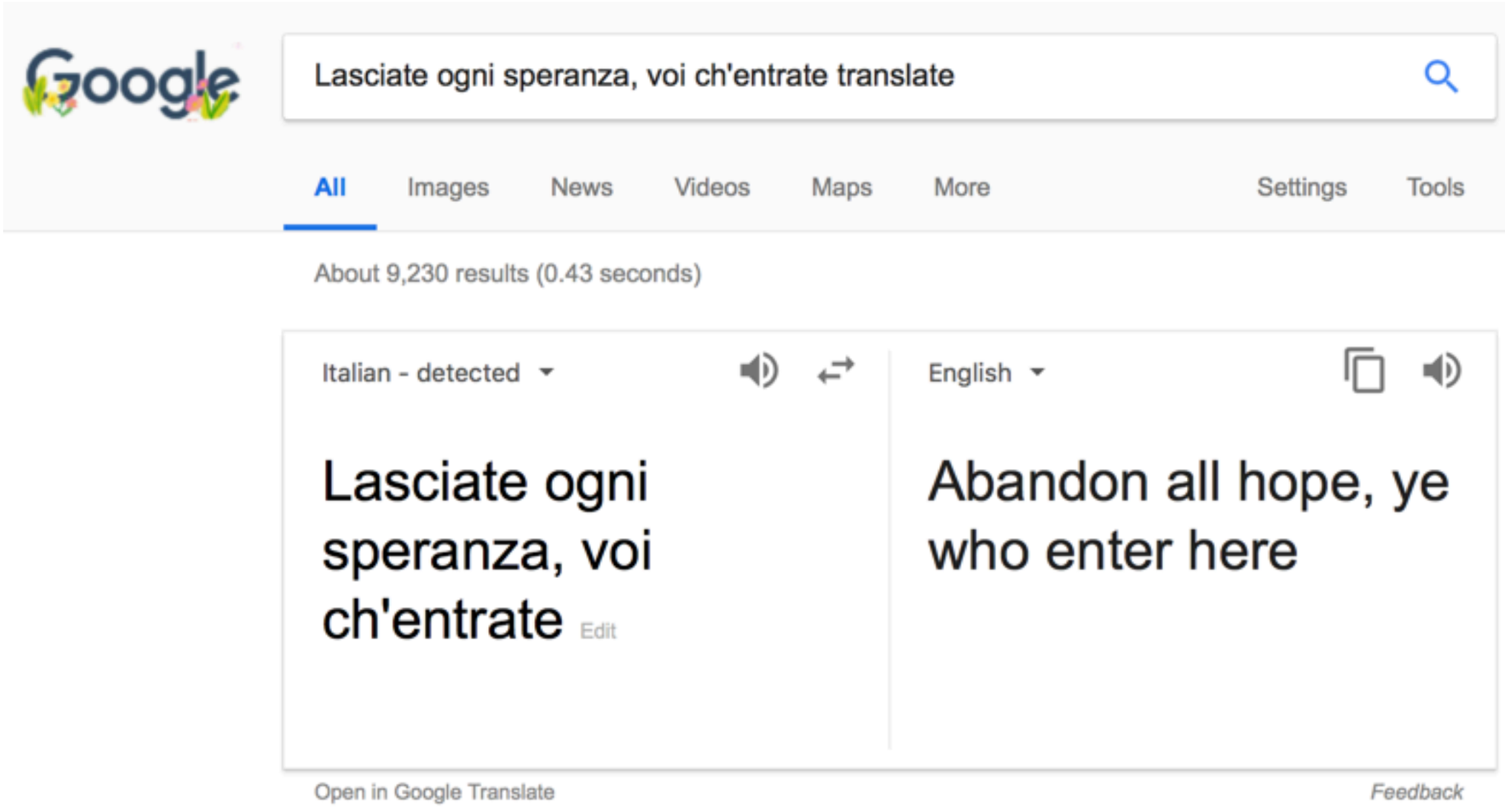
Character LM

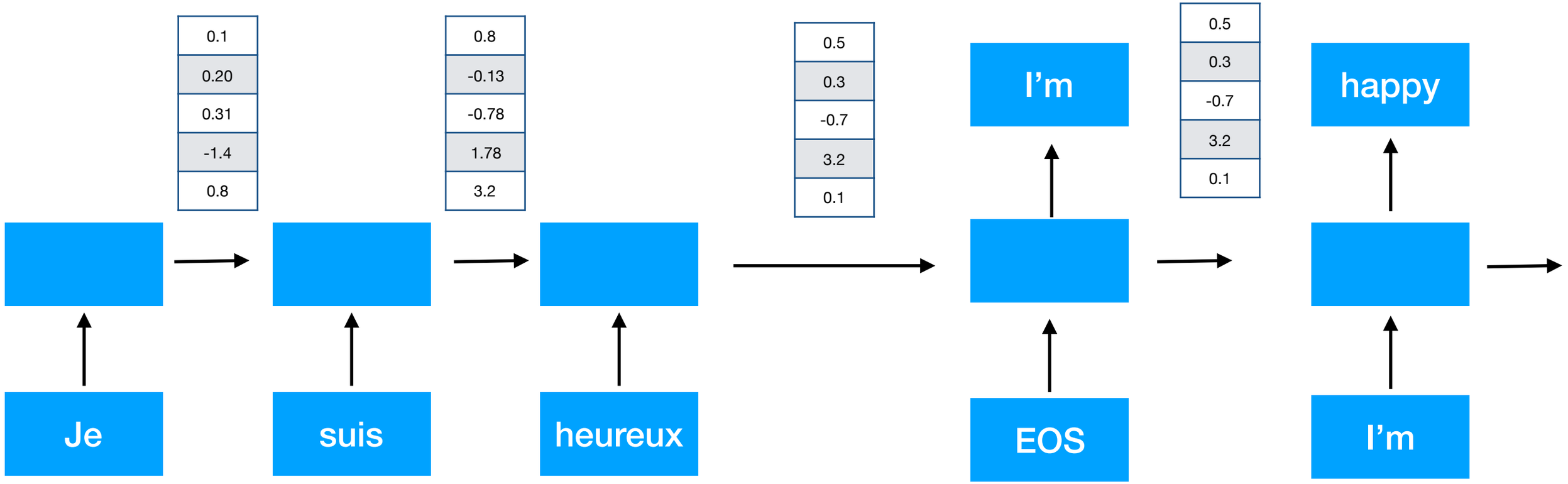
```
PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.
```

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Recurrent neural network

- Often used for sequential prediction tasks:
    - Language models—predicting the next symbol (word, character) in a sequence
    - Machine translation—-predicting a sequence of words (sentence) in language *f* conditioned on sentence in language *e*

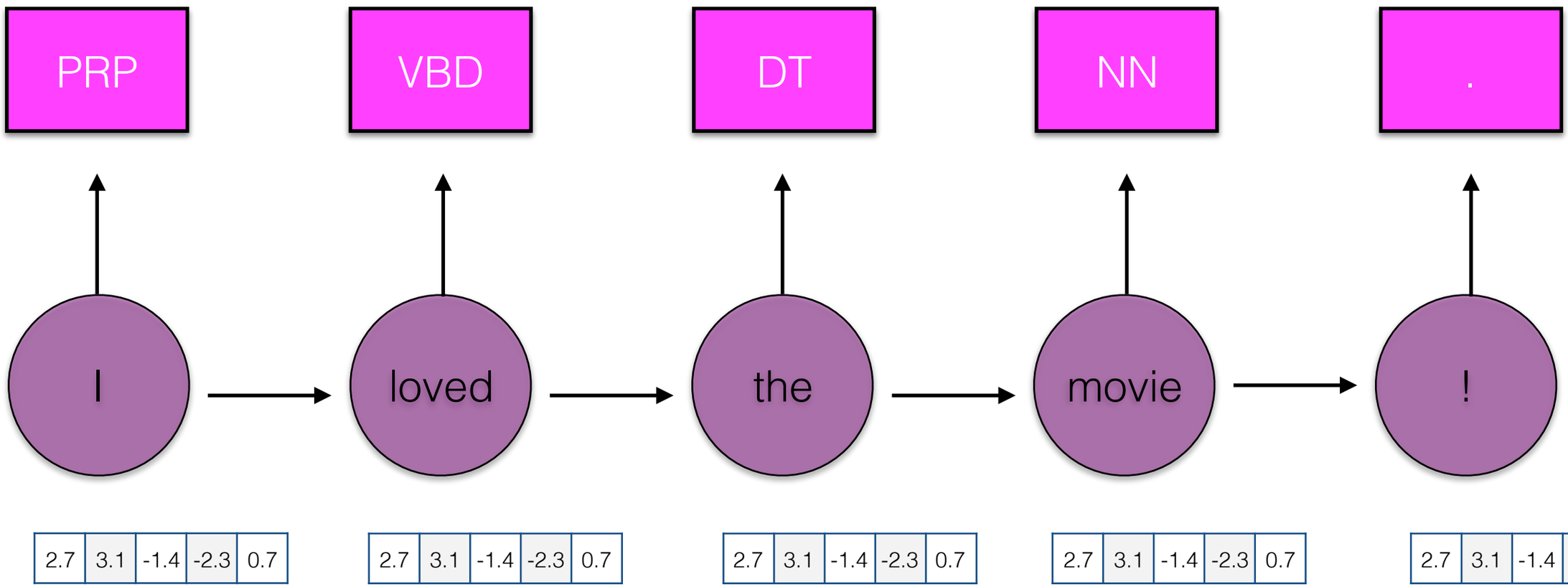

Google
Lasciate ogni speranza, voi ch'entrate translate
All
Images
News
Videos
Maps
More
Settings
Tools
About 9,230 results (0.43 seconds)
Italian - detected
English
Lasciate ogni speranza, voi ch'entrate Edit
Abandon all hope, ye who enter here
Open in Google Translate
Feedback


0.1
0.20
0.31
-1.4
0.8
0.8
-0.13
-0.78
1.78
3.2
0.5
0.3
-0.7
3.2
0.1
I’m
0.5
0.3
-0.7
3.2
0.1
happy
Je
suis
heureux
EOS
I’m

Recurrent neural network

- Often used for sequential prediction tasks:
    - Language models—predicting the next symbol (word, character) in a sequence
    - Machine translation—-predicting a sequence of words (sentence) in language *f* conditioned on sentence in language *e*
    - Sequence labeling (POS tagging, NER)—we'll cover that after spring break.

For POS tagging, predict the tag from $\boldsymbol{y}$ conditioned on the context

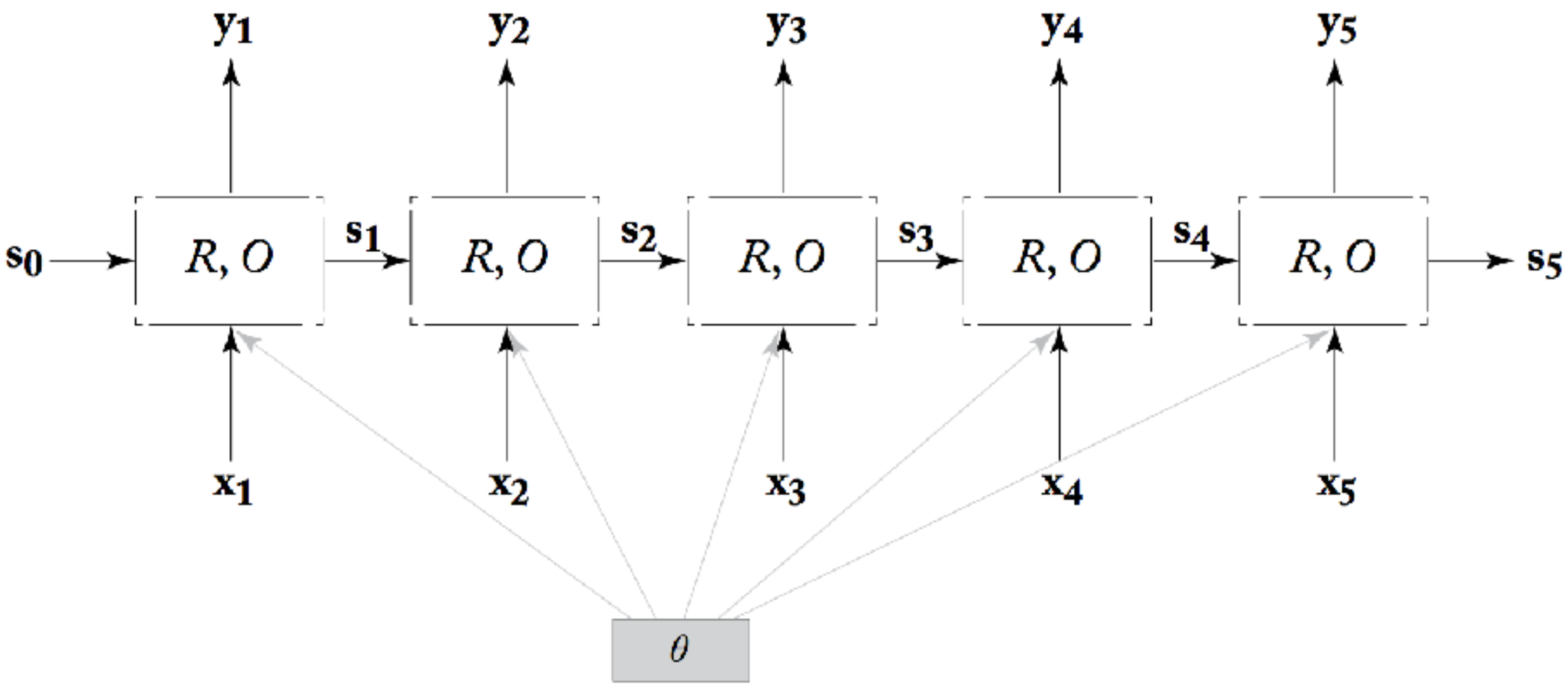


Recurrent neural network

- We'll use them today to generate a representation of a *sequence* of arbitrary length (sentence, document, etc.) optimized for some task.

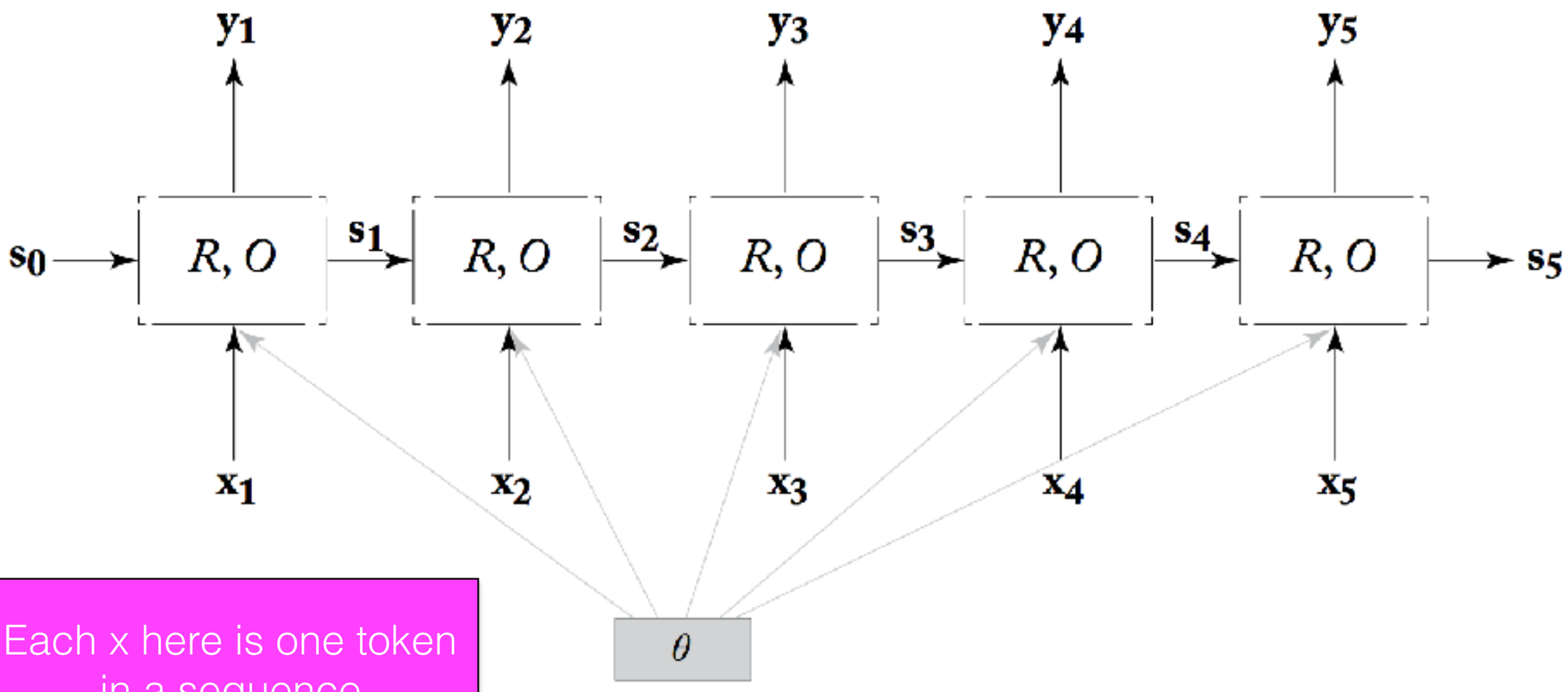
- We can then use that representation for e.g. text classification/regression.

Recurrent neural network



Goldberg 2017

Each y is the output of the RNN at that time step; sometimes we use this information (POS tagging, LM); sometimes we only use the output for the final state (s_5)

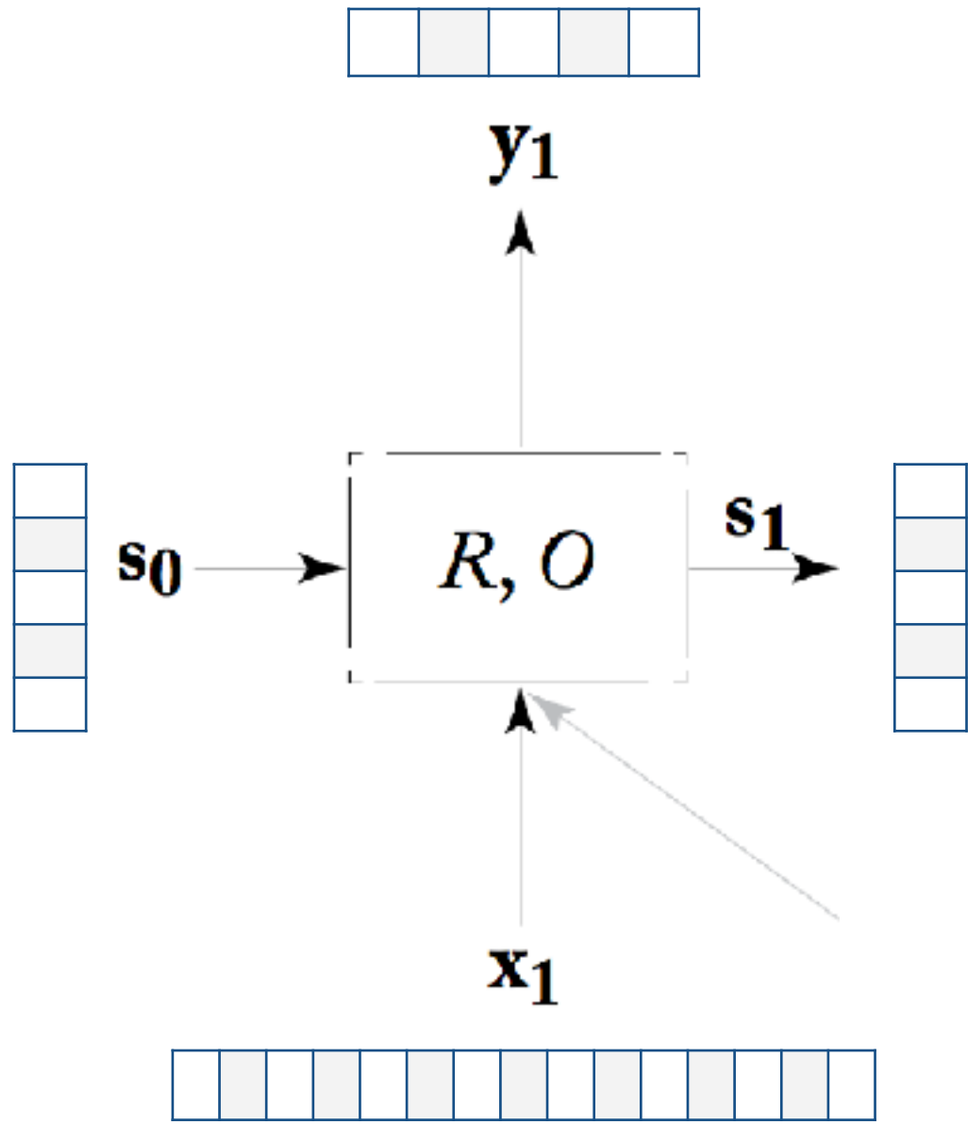


Each x here is one token in a sequence

Goldberg 2017

Recurrent neural network

- Each time step has two inputs:
    - x_i (the observation at time step i); one-hot vector, feature vector or word embedding.
- s_{i-1} (the output of the previous state); base case: s_0 = 0 vector



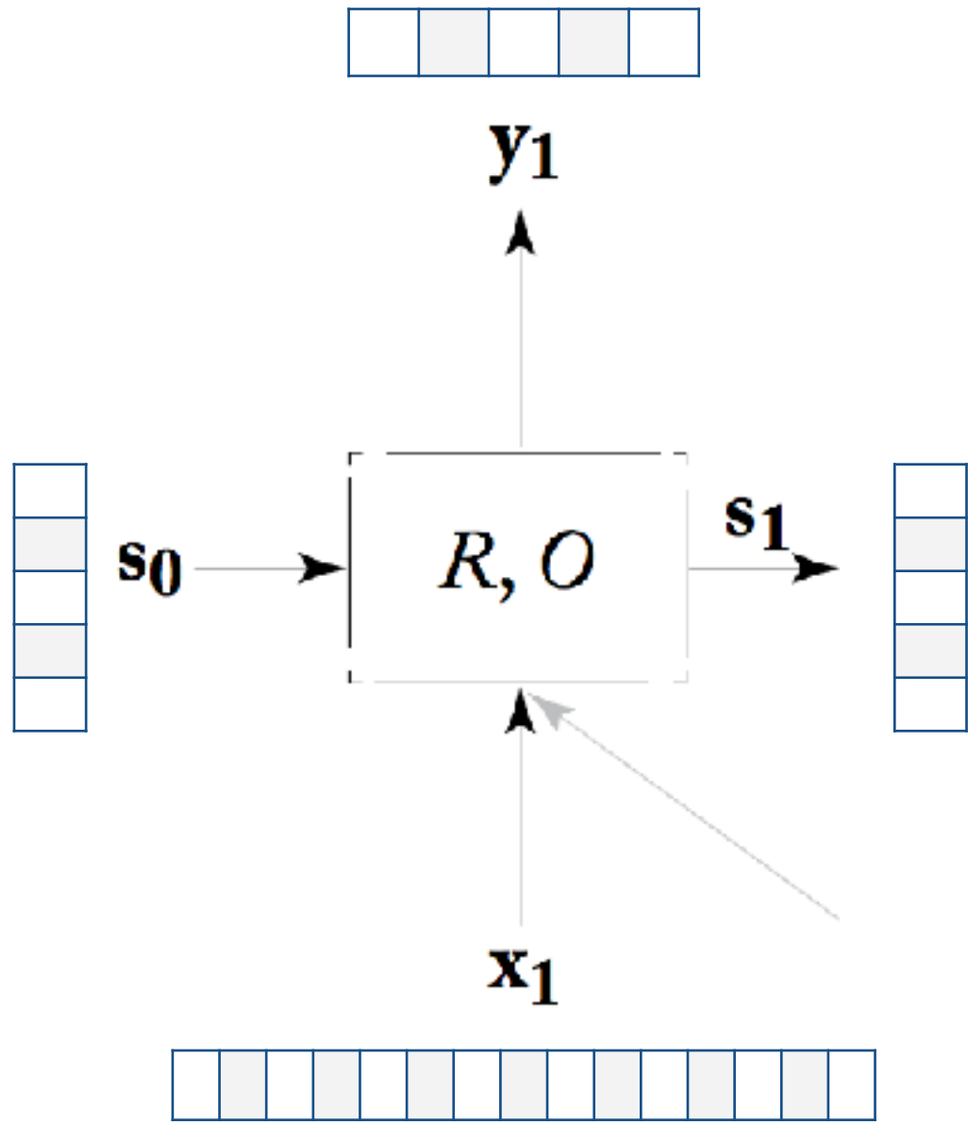
Recurrent neural network

$$s_i = R(x_i, s_{i-1})$$

R computes the output state as a function of the current input and previous state

$$y_i = O(s_i)$$

O computes the output as a function of the current output state



“Simple” RNN

g = tanh or relu

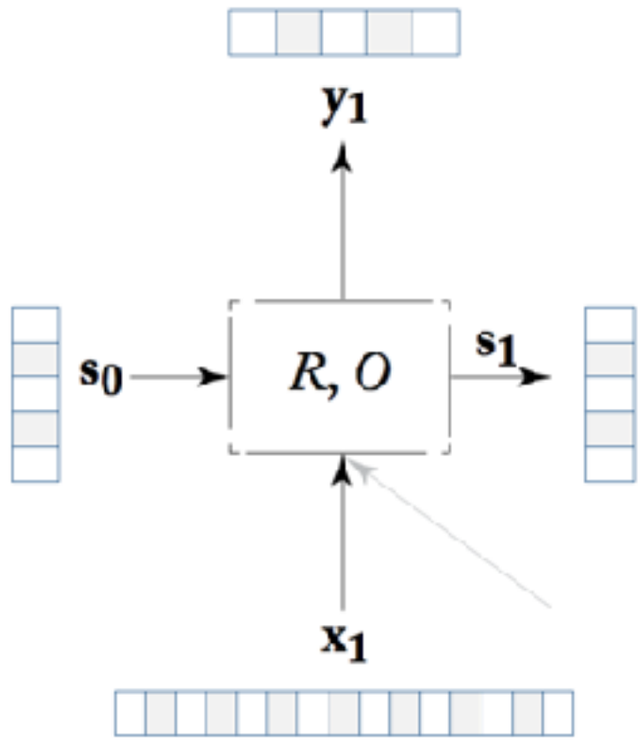
$$s_i = R(x_i, s_{i-1}) = g(s_{i-1}W^s + x_iW^x + b)$$

Different weight vectors W transform the previous state and current input before combining

$$W^s \in \mathbb{R}^{H \times H}$$

$$W^x \in \mathbb{R}^{D \times H}$$

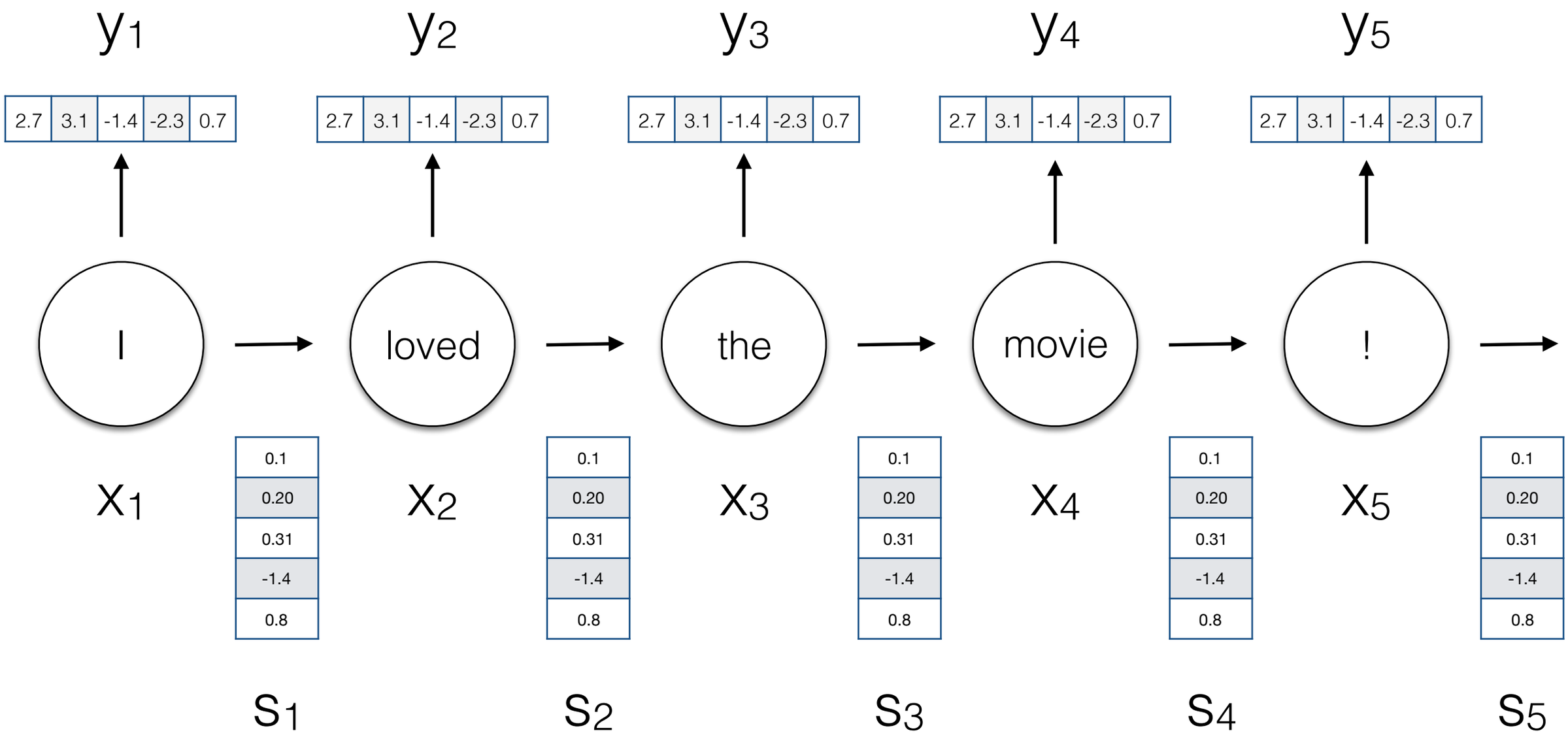
$$b \in \mathbb{R}^H$$

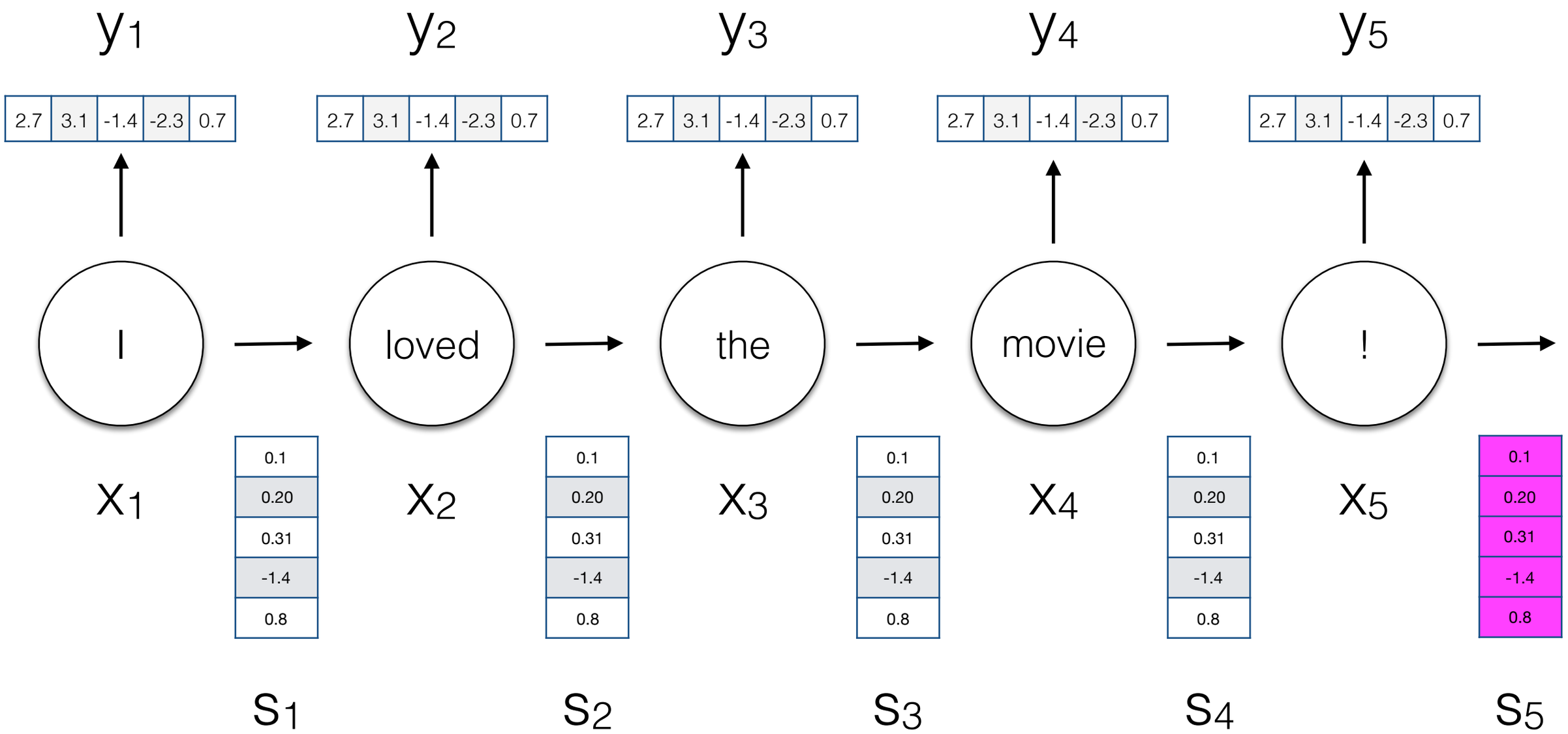


$$y_i = O(s_i) = s_i$$

Elman 1990, Mikolov 2012

How do we use RNNs for document classification?

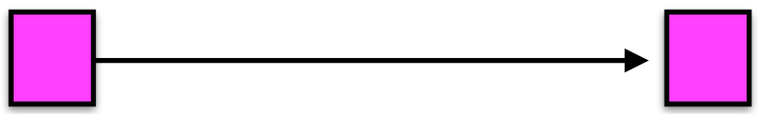

y1
y2
y3
y4
y5
2.7 3.1 -1.4 -2.3 0.7
I
loved
the
movie
!
x1
x2
x3
x4
x5
0.1
0.20
0.31
-1.4
0.8
s1
s2
s3
s4
s5


y1
y2
y3
y4
y5
2.7 3.1 -1.4 -2.3 0.7
I
loved
the
movie
!
x1
x2
x3
x4
x5
0.1 0.20 0.31 -1.4 0.8
s1
s2
s3
s4
s5

RNNs

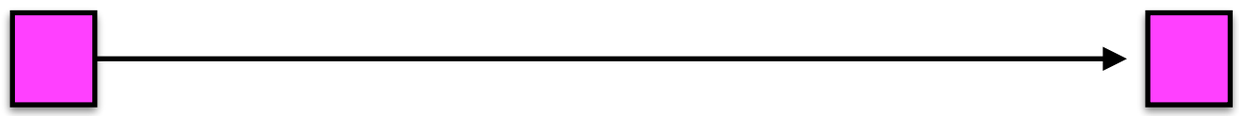
- The final hidden state contains a representation informed by the entire sequence, including non-linear interactions between the elements of that sequence.

If your date likes it , do not date that person again



in conditional until comma, so downplay “likes”

The director said it was “ the best movie he ever made ”



quoted speech is in between quotation marks, so downplay “best”

RNNs

- Recurrent networks are deep in that they involve one “layer” for each time step (e.g., words in a sentence)

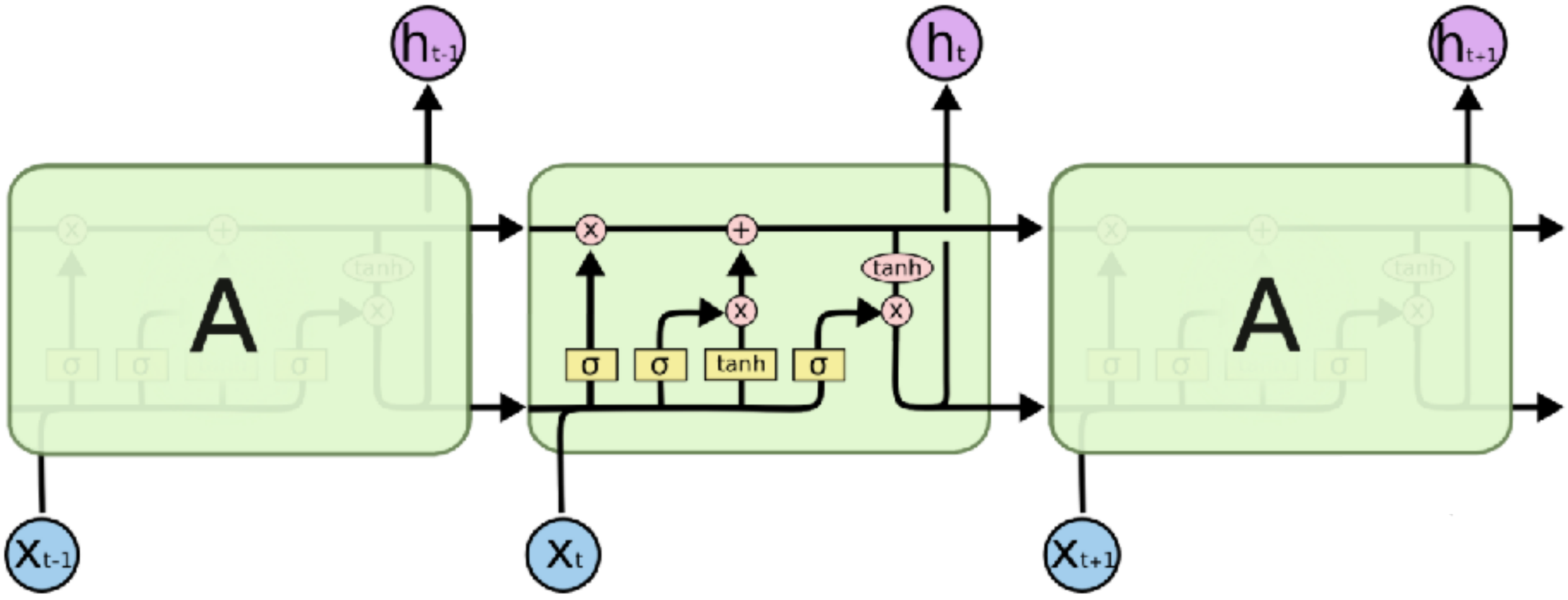
- Vanishing gradient problem: as error is back propagated through the layers of a deep network, they tend toward 0.

Long short-term memory network (LSTM)

- Designed to account for the vanishing gradient problem

- Basic idea: split the *s* vector propagated between time steps into a memory component and a hidden state component

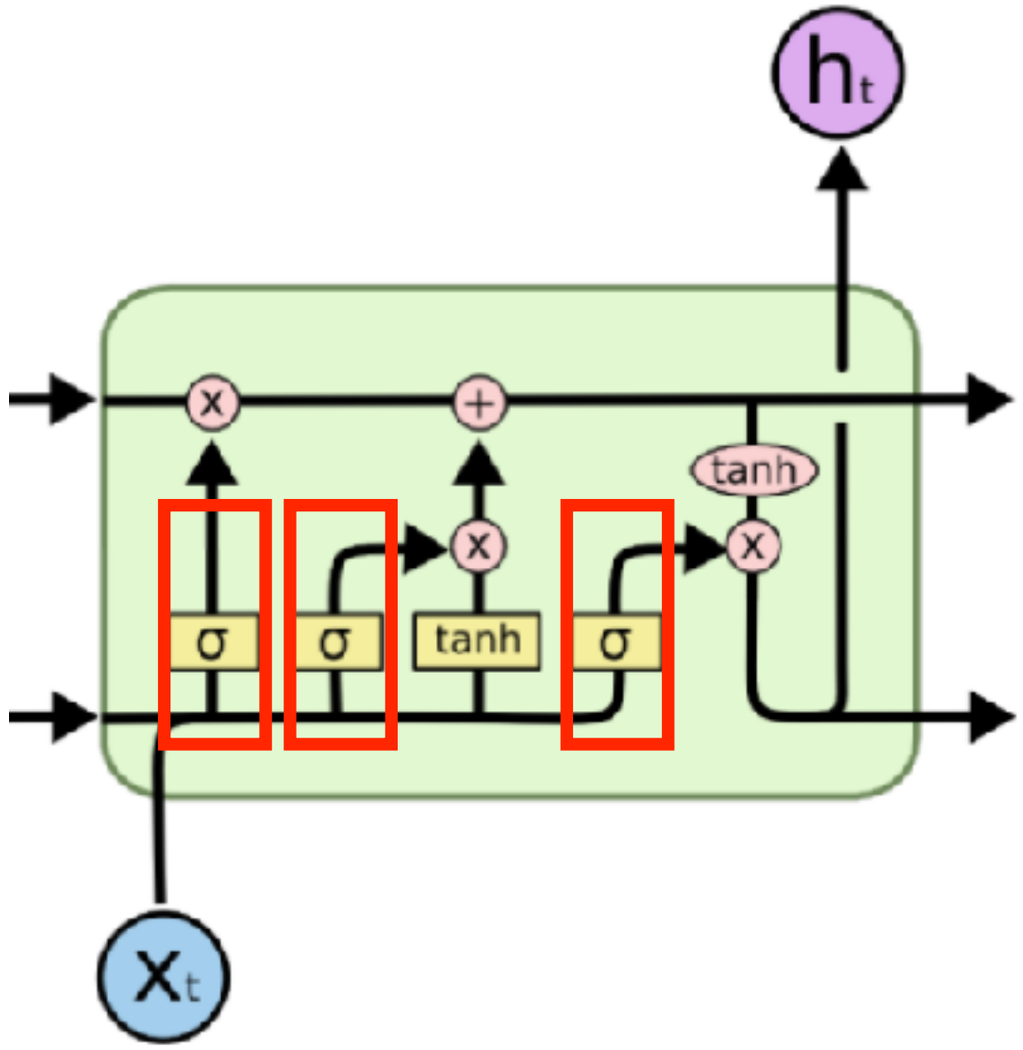
LSTMs



http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Gates

- LSTMs gates control the flow of information



- A sigmoid squashes its input to between 0 and 1
- By multiplying the output of a sigmoid elementwise with another vector, we forget information in the vector (if multiplied by a number close to 0) or allow it to pass (if multiplied by number close to 1)

Forget gate: as a function of the previous hidden state and current input, forget information in the memory

3.7	1.4	-0.7	-1.4	7.8

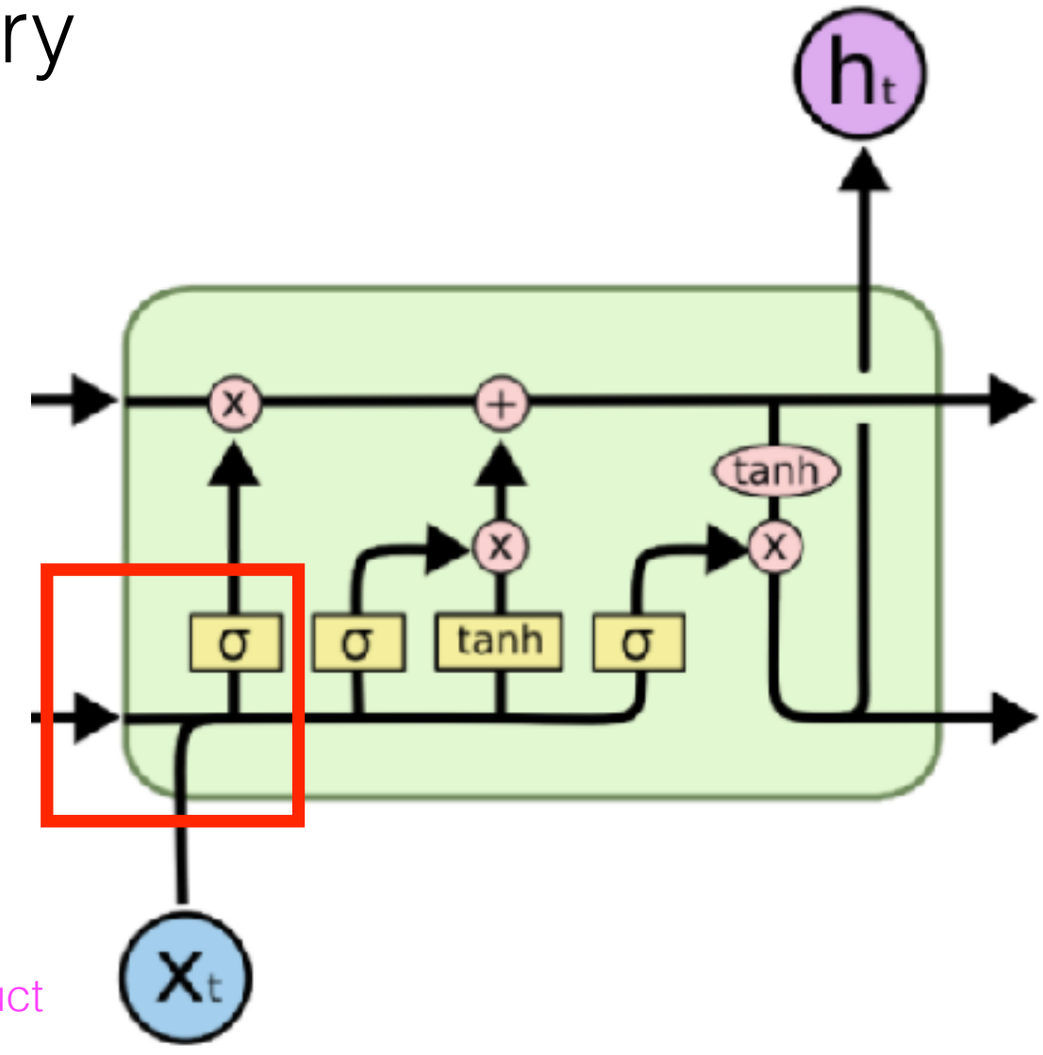
memory

0.01	0.99	0.5	0.98	0.01

hidden state

0.03	1.4	-0.35	-1.38	0.08

elementwise product



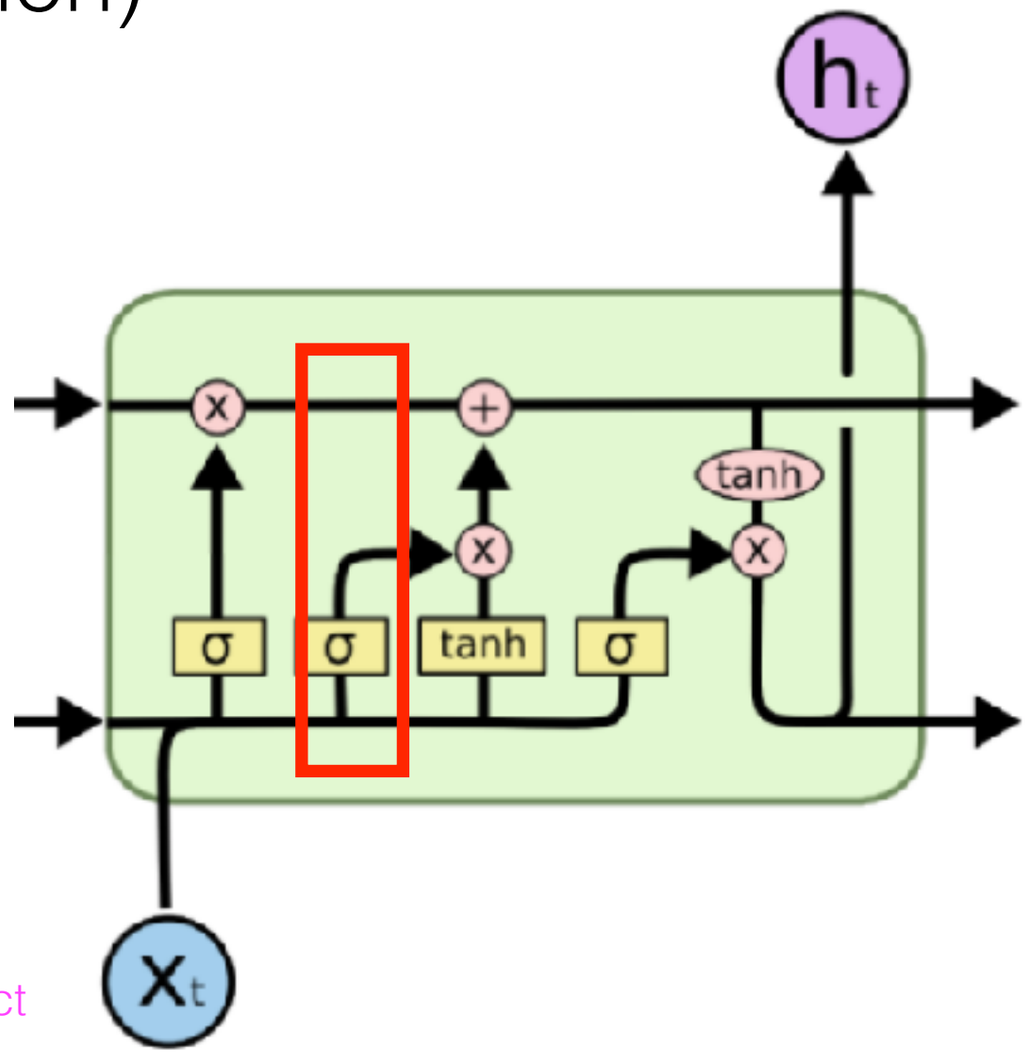
Input gate (but forget some information about the current observation)

0.03	1.4	-0.35	-1.38	0.08

memory

hidden state

elementwise product



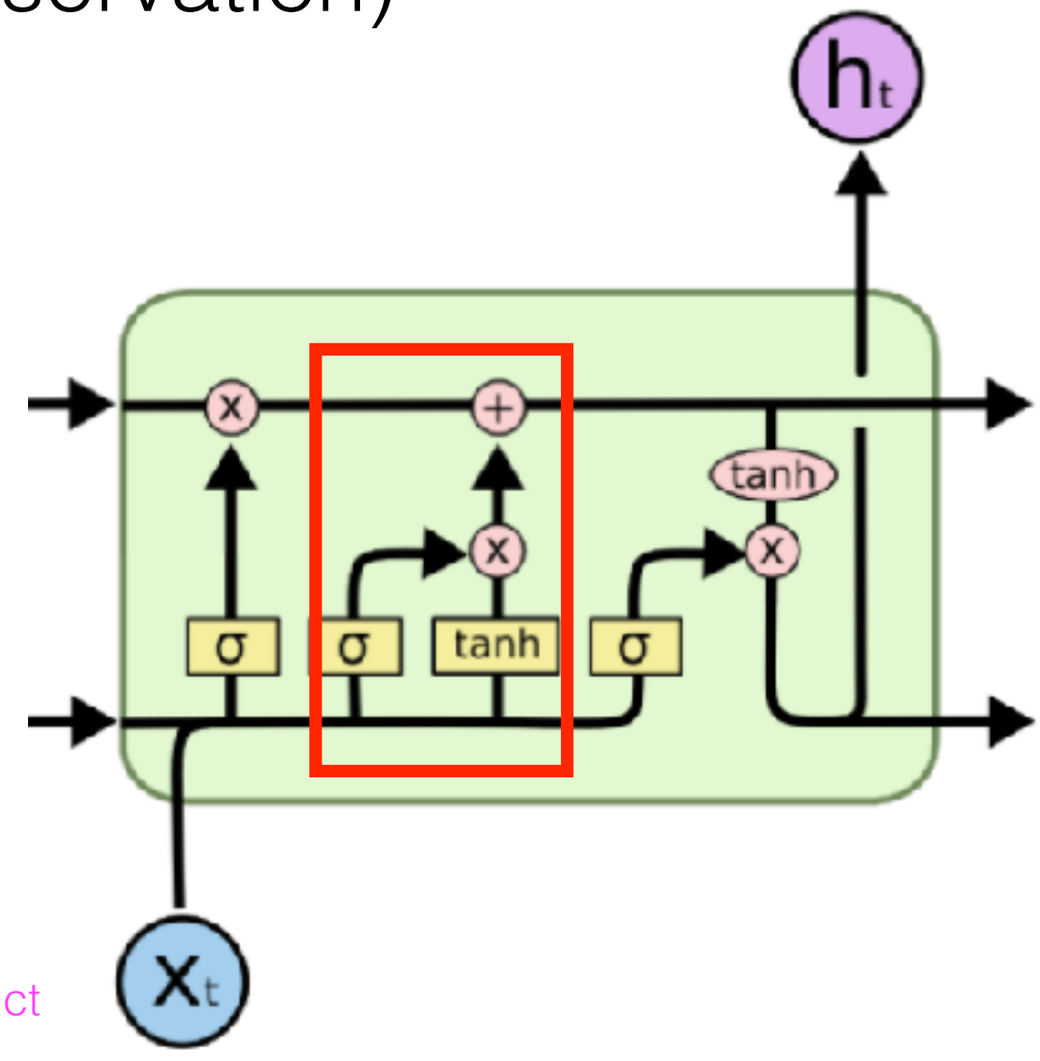
Update the memory (but forget some information about the current observation)

0.03	1.4	-0.35	-1.38	0.08

memory

hidden state

elementwise product



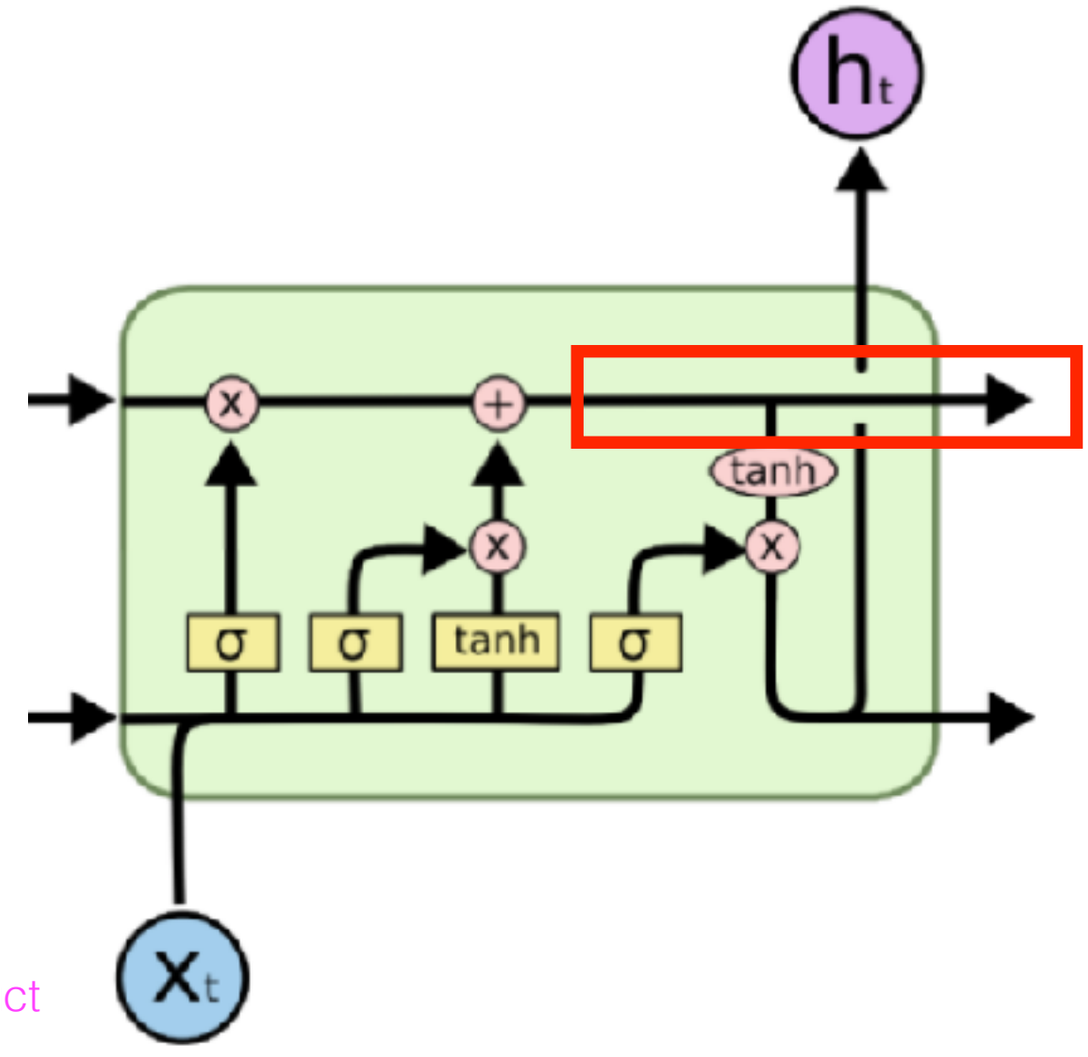
The memory passes directly to the next state

0.8	0.4	-1.4	9.8	3.4

memory

hidden state

elementwise product

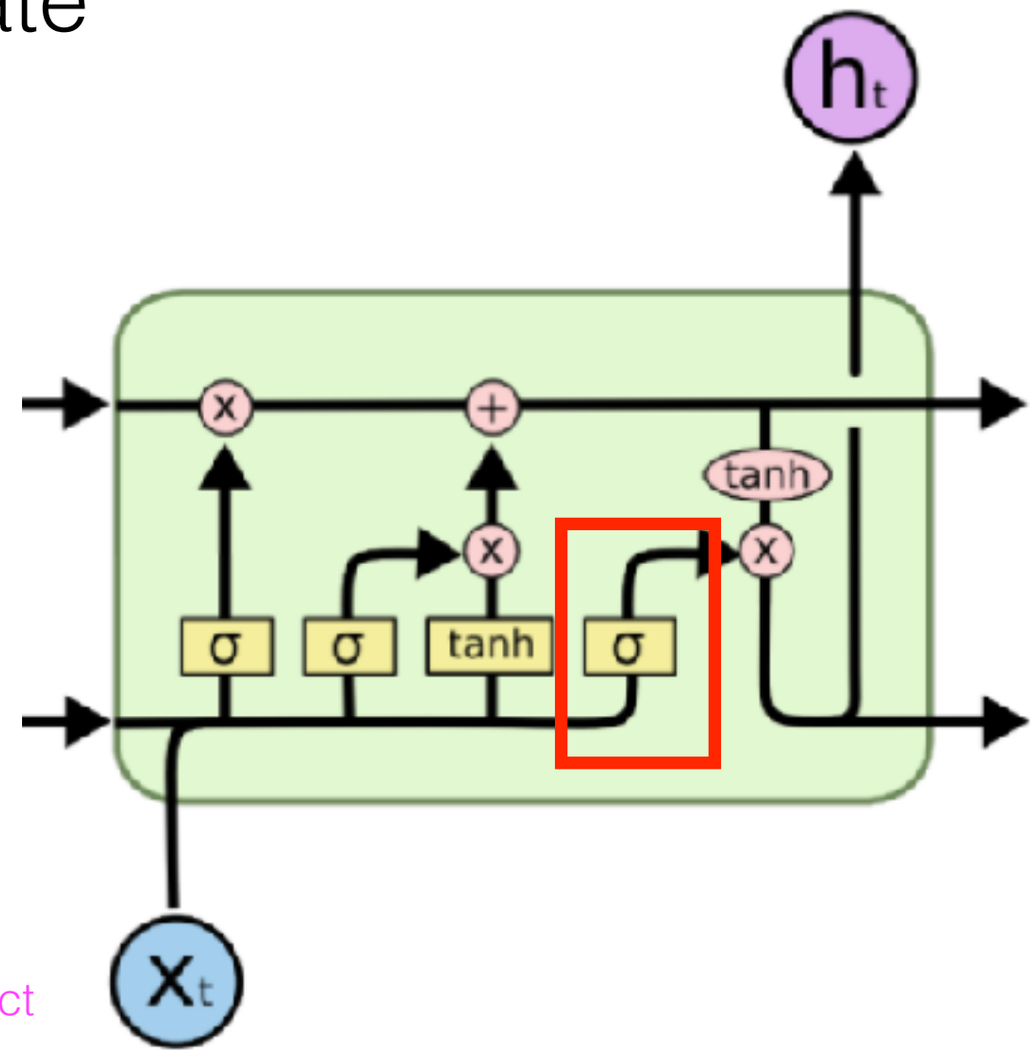


Output gate: forget some information
to send to the hidden state

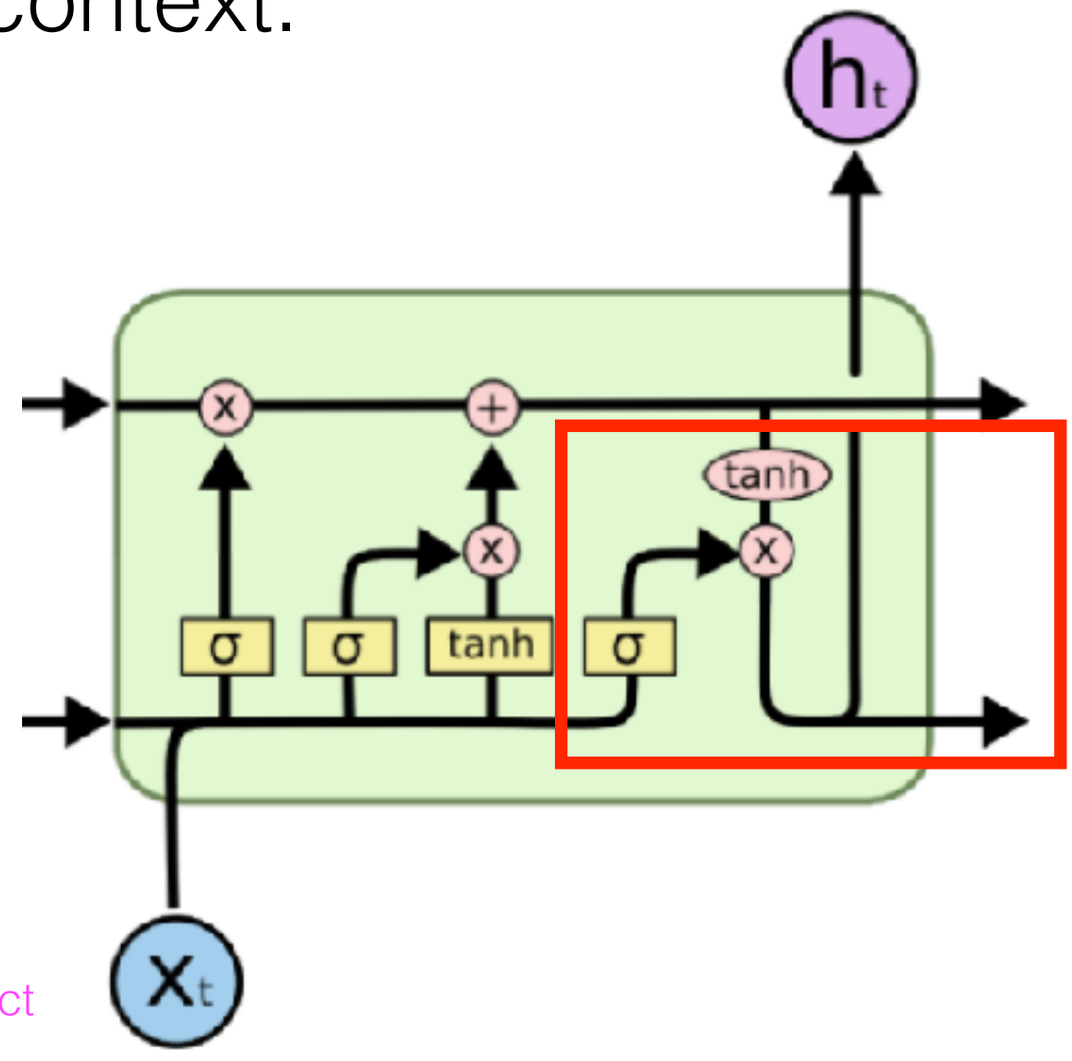
memory

hidden state

elementwise product



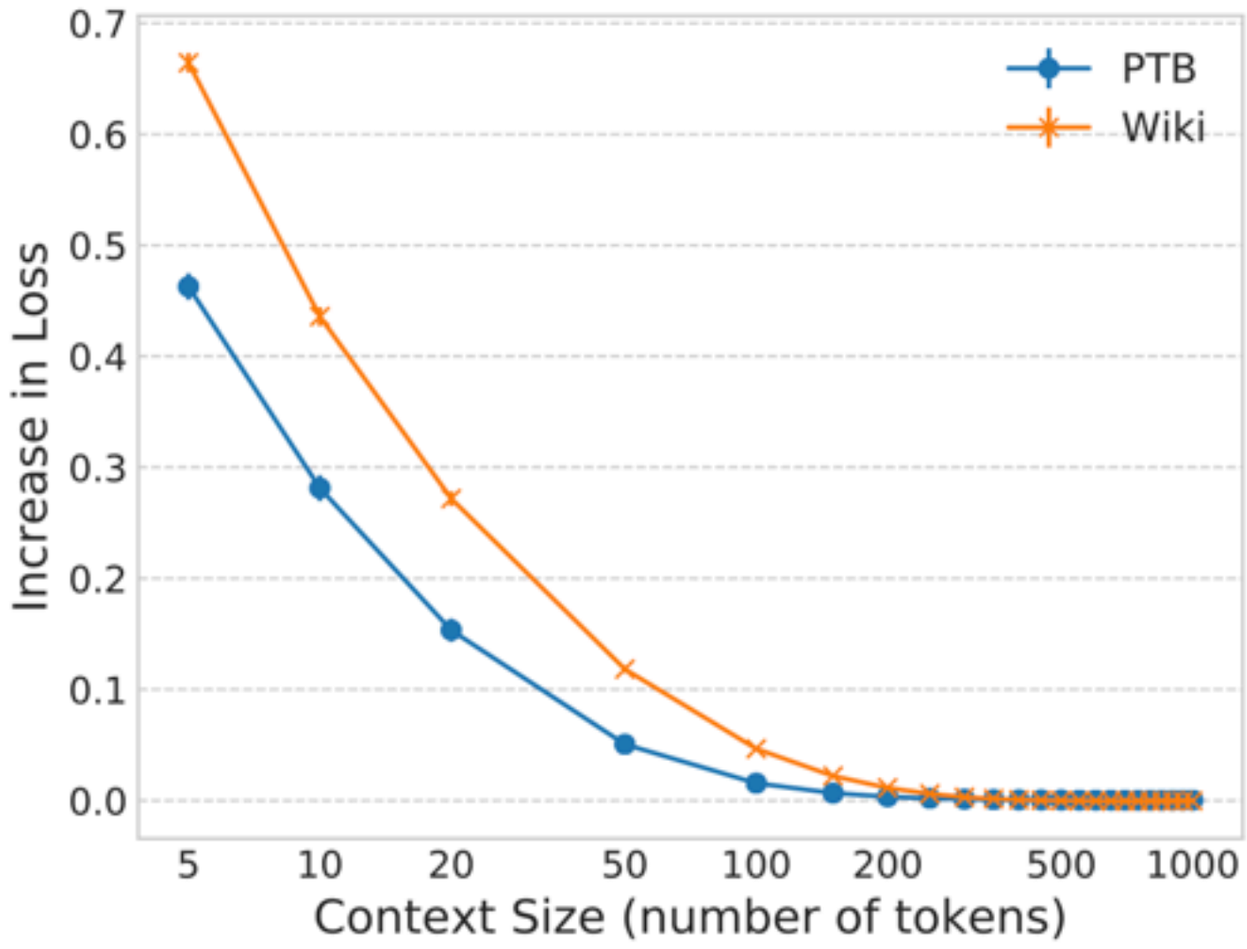
The hidden state is updated with the current observation and new context.



How much context?

- For language modeling, LSTMs are aware of about 200 words of context

- Ignores word order beyond 50 words

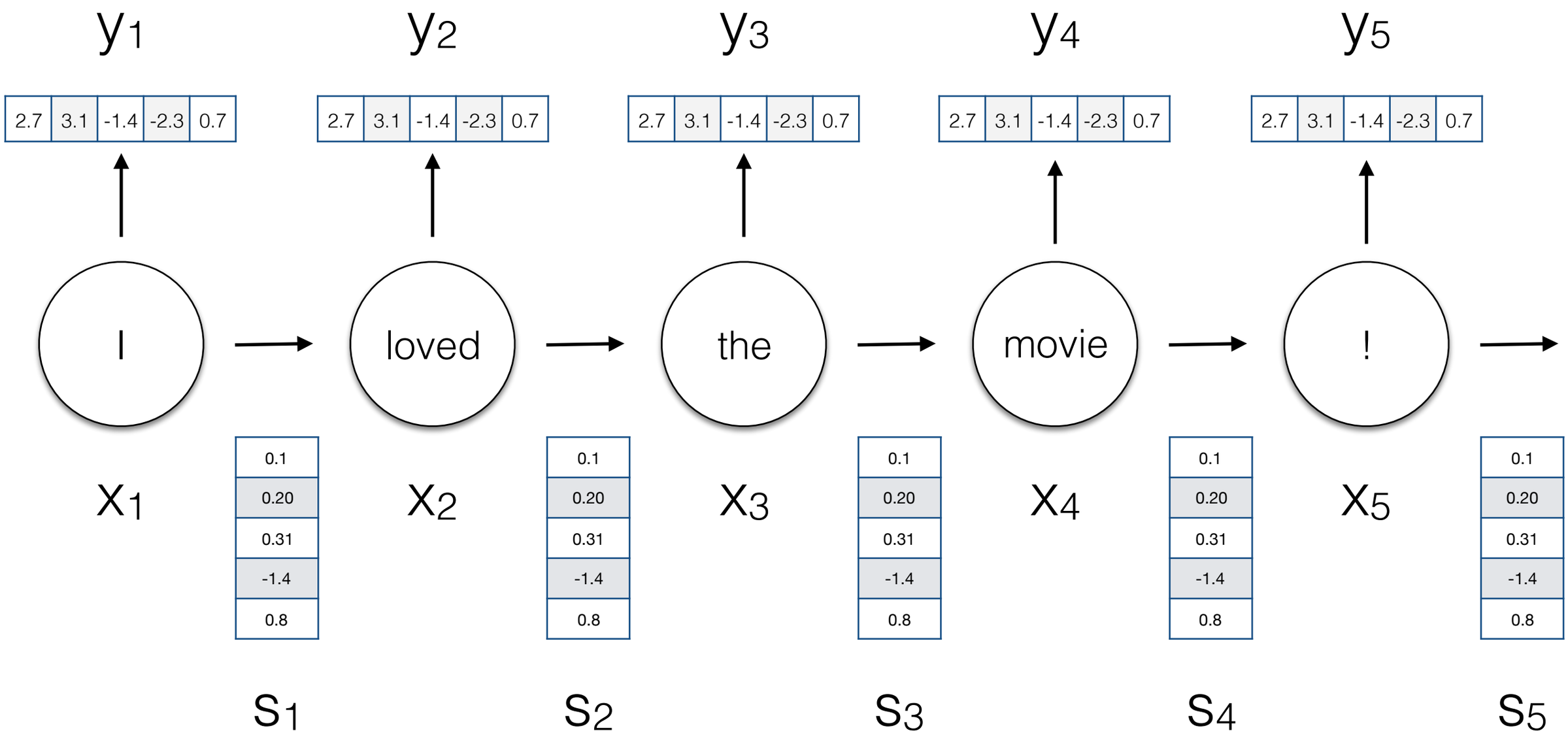


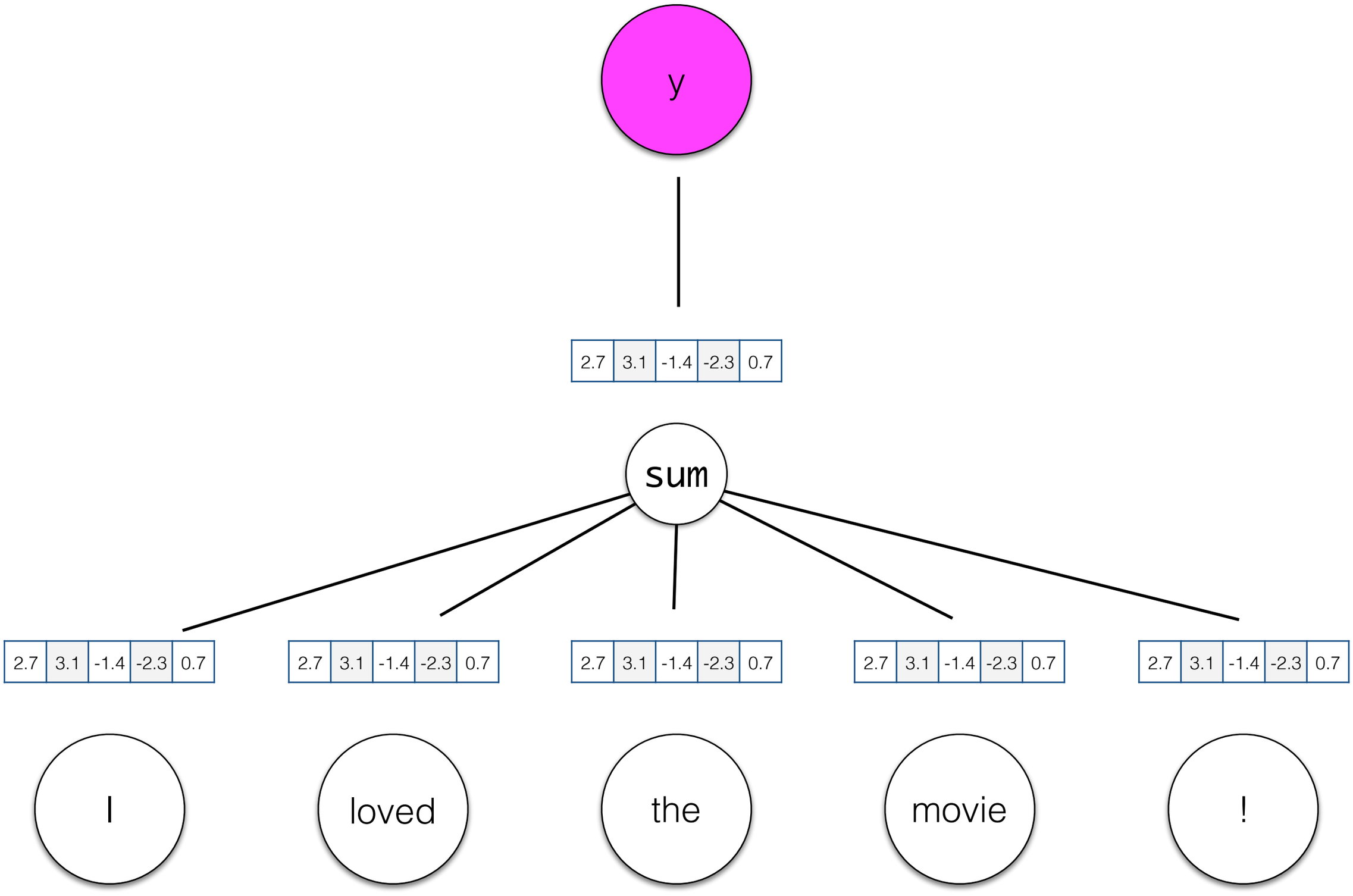
Khandelwal et al. (2018), “Sharp Nearby, Fuzzy Far Away: How Neural Language Models Use Context” (ACL)

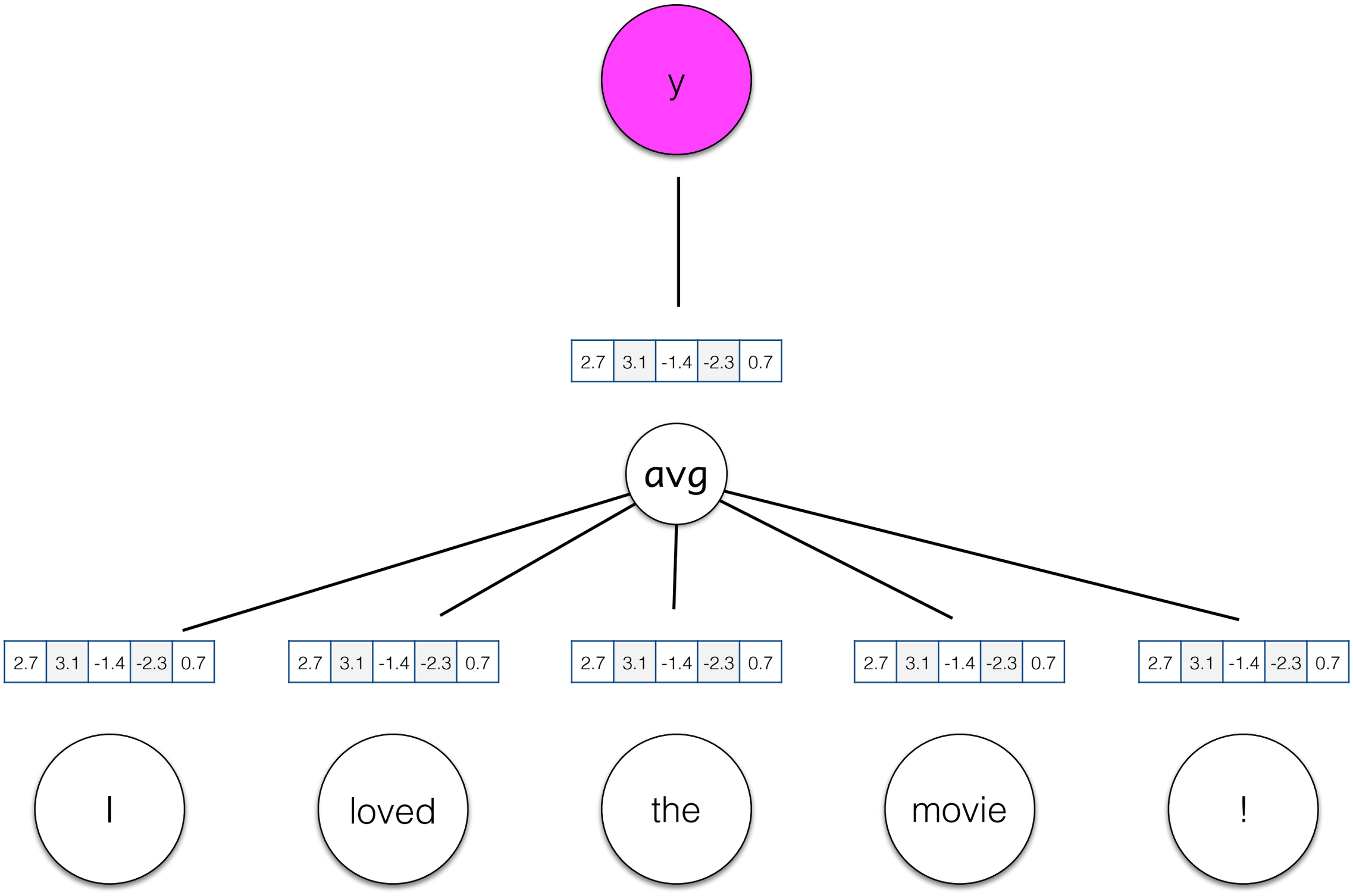
Context

- Encoding an entire sequence into a fixed dimensional vector at the end of the sequence can potentially lose a lot of information, especially for documents.

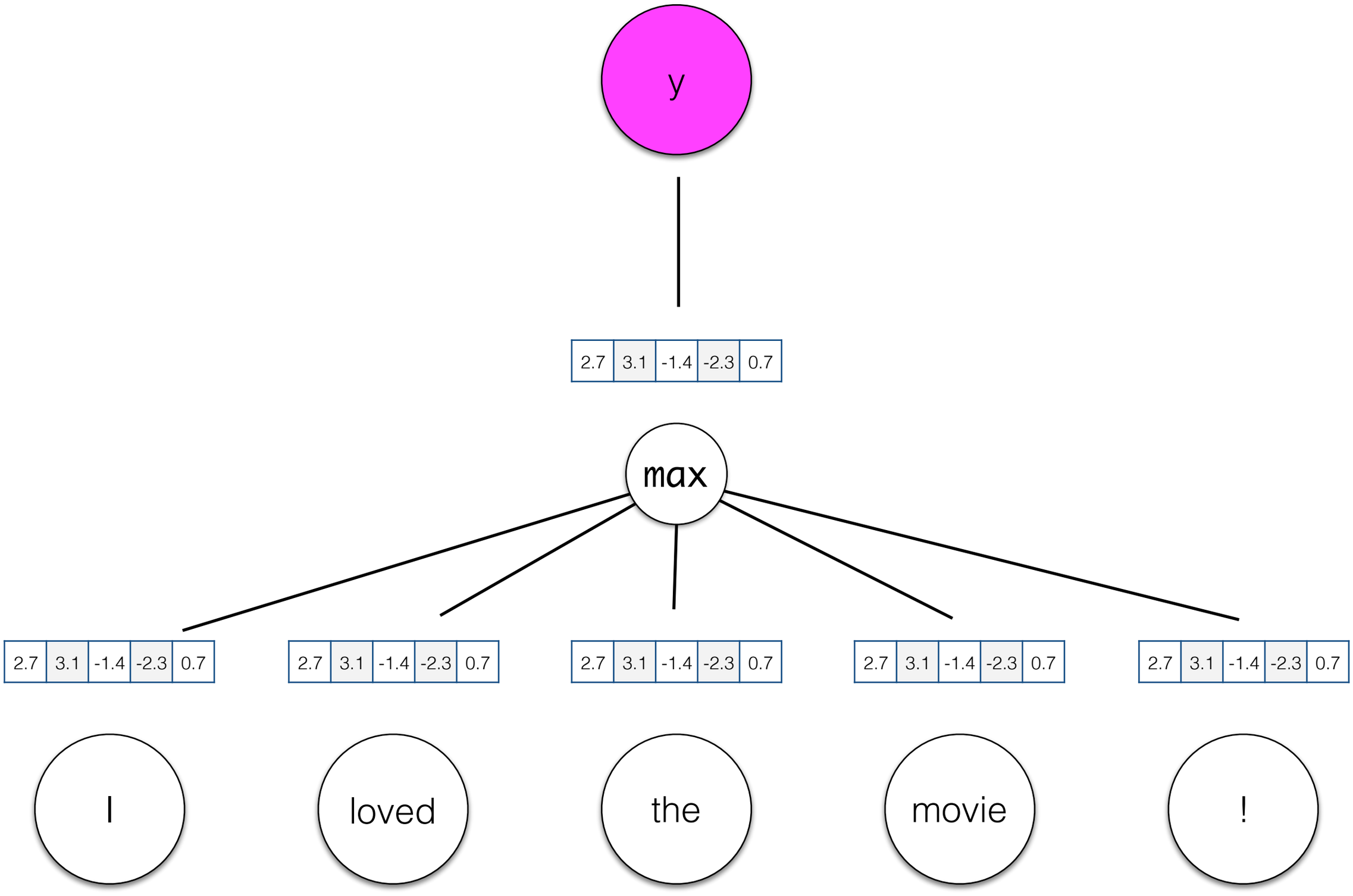
- Rarely used in practice.


y1
y2
y3
y4
y5
2.7 3.1 -1.4 -2.3 0.7
I
loved
the
movie
!
x1
x2
x3
x4
x5
0.1
0.20
0.31
-1.4
0.8
s1
s2
s3
s4
s5


y
2.7
3.1
-1.4
-2.3
0.7
sum
2.7
3.1
-1.4
-2.3
0.7
2.7
3.1
-1.4
-2.3
0.7
2.7
3.1
-1.4
-2.3
0.7
2.7
3.1
-1.4
-2.3
0.7
I
loved
the
movie
!



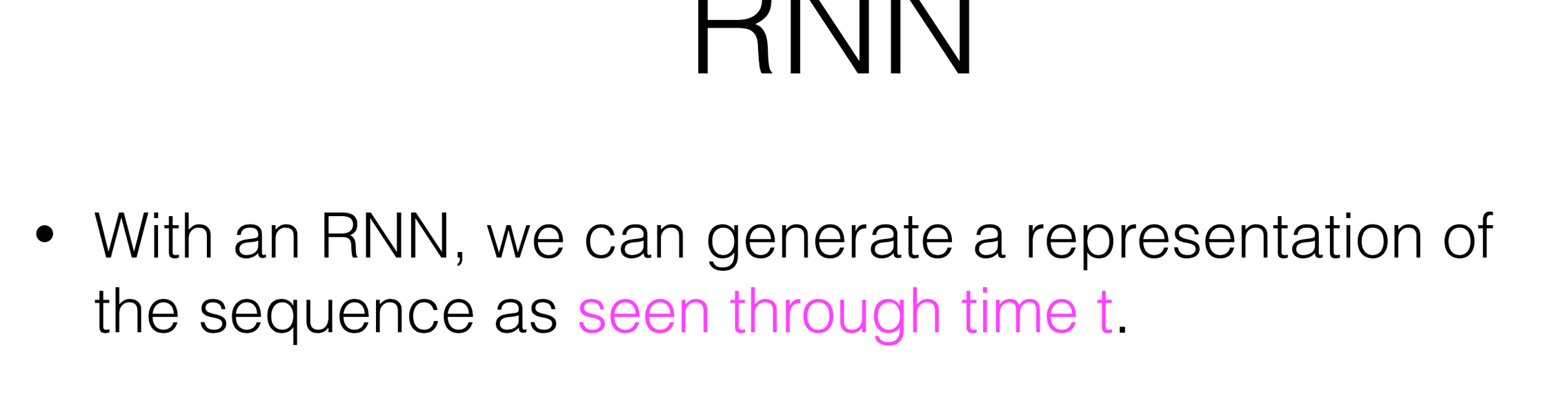
Iyyer et al. (2015), “Deep Unordered Composition Rivals Syntactic Methods for Text Classification” (ACL)


y
2.7
3.1
-1.4
-2.3
0.7
max
2.7
3.1
-1.4
-2.3
0.7
2.7
3.1
-1.4
-2.3
0.7
2.7
3.1
-1.4
-2.3
0.7
2.7
3.1
-1.4
-2.3
0.7
I
loved
the
movie
!

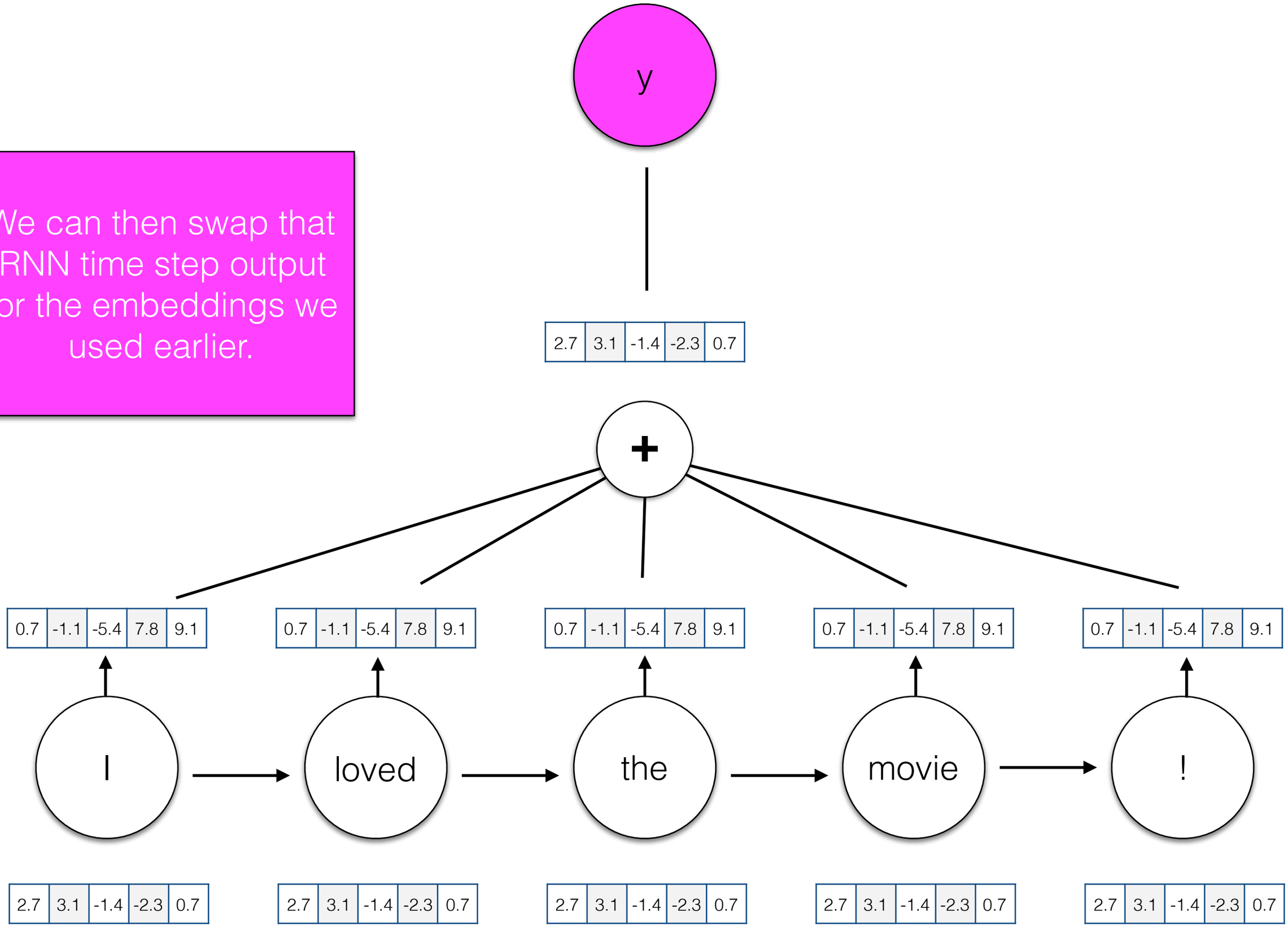
RNN

- With an RNN, we can generate a representation of the sequence as seen through time t.

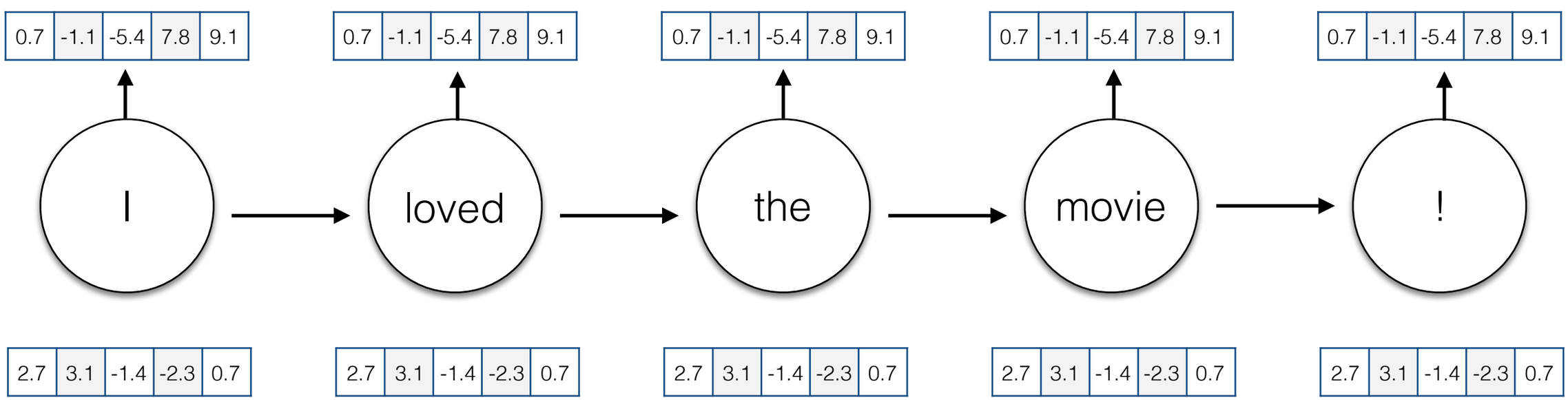
- This encodes a representation of meaning specific to the local context a word is used in.



We can then swap that RNN time step output for the embeddings we used earlier.



What about the future context?



Bidirectional RNN

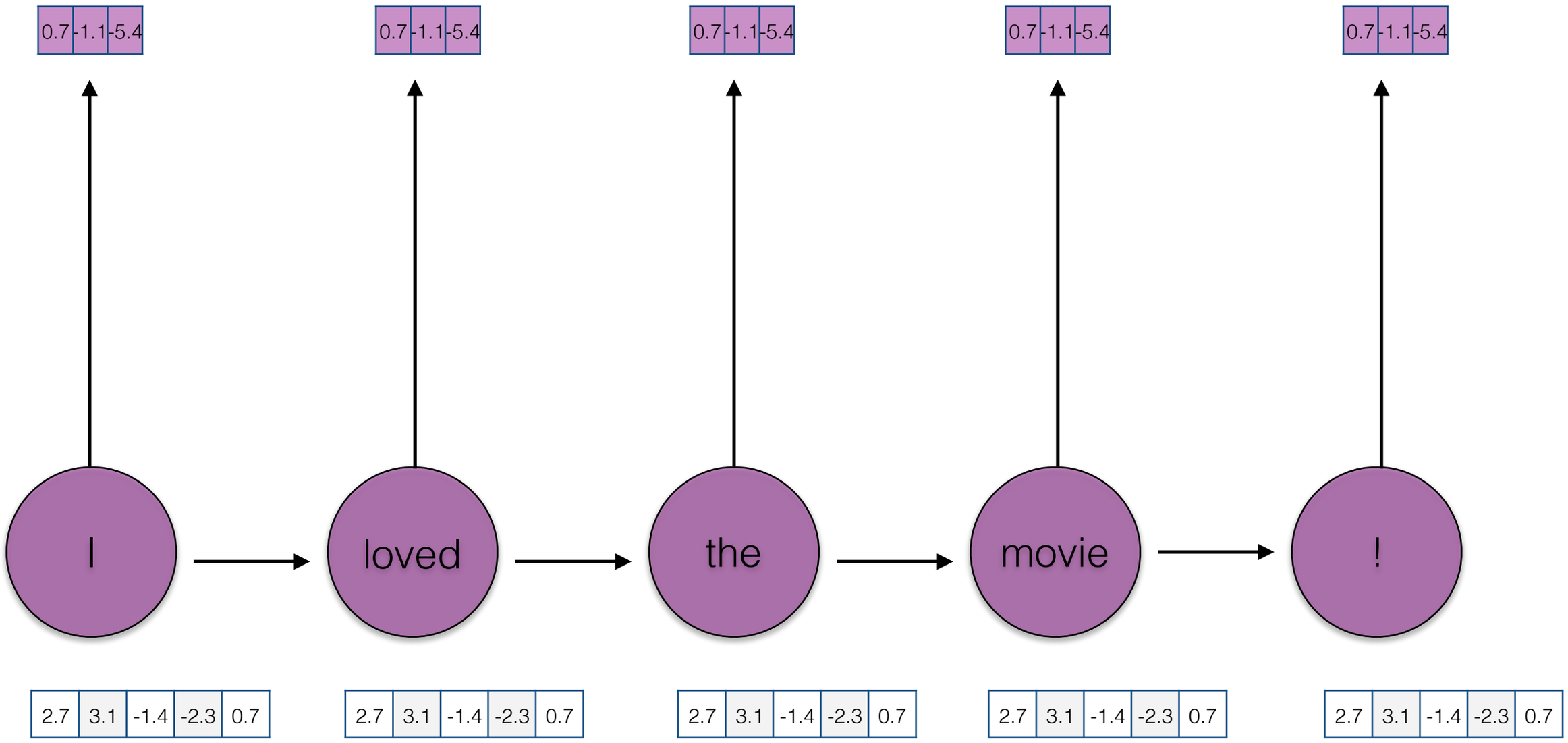
- A powerful alternative is make predictions conditioning both on the past and the future.

- Two RNNs
    - One running left-to-right
    - One right-to-left

- Each produces an output vector at each time step, which we concatenate

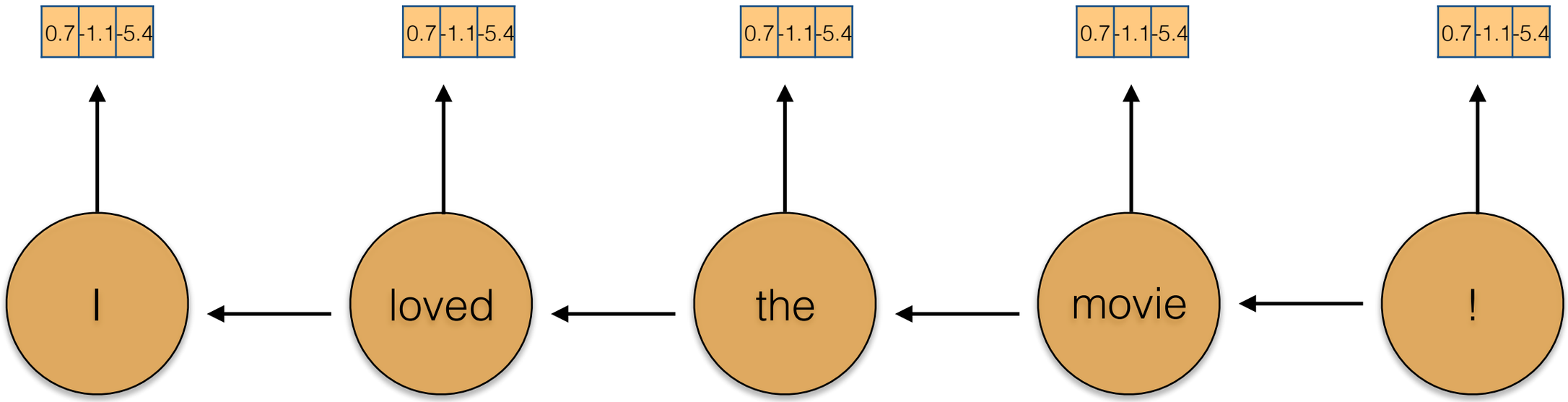
Bidirectional RNN

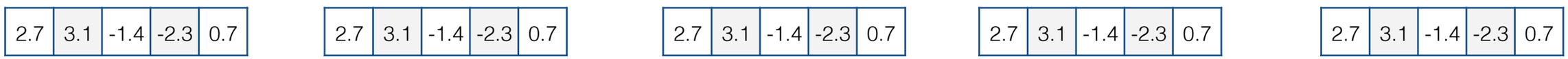
forward RNN



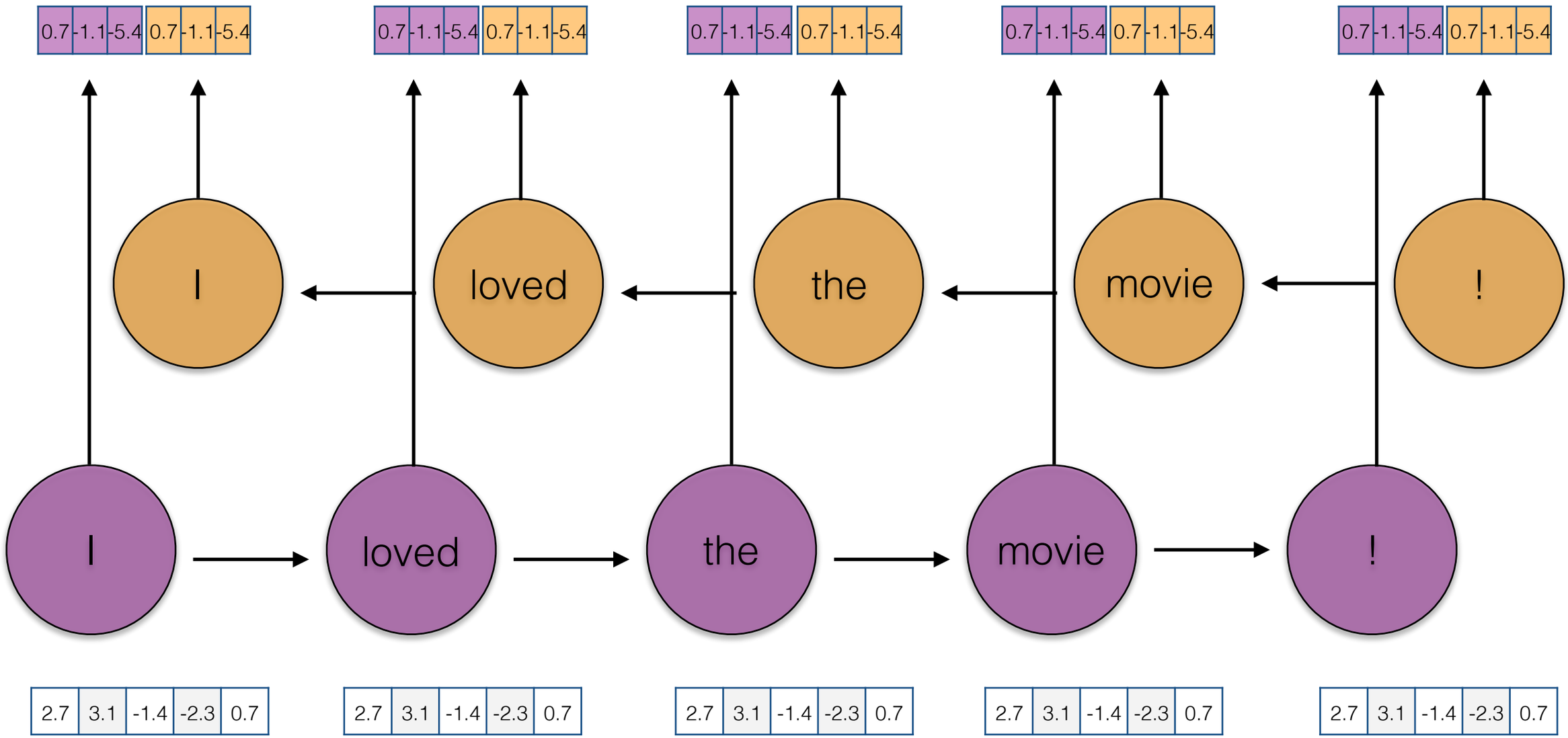
Bidirectional RNN

backward RNN





Bidirectional RNN



Bidirectional RNN

- The forward RNN and backward RNN each output a vector of size H at each time step, which we concatenate into a vector of size 2H.

- The forward and backward RNN each have separate parameters to be learned during training.

Training BiRNNs

- Given this definition of an BiRNN:

$$s_b^i = R_b(x^i, s_b^{i+1}) = g(s_b^{i+1} W_b^s + x^i W_b^x + b_b)$$

$$s_f^i = R_f(x^i, s_f^{i-1}) = g(s_f^{i-1} W_f^s + x^i W_f^x + b_f)$$

$$y_i = \text{softmax}\left([s_f^i; s_b^i] W^o + b^o\right)$$

- We have 8 sets of parameters to learn (3 for each RNN + 2 for the final layer)

Padding

- Many neural network libraries require each sequence within the same batch to be the same length.

- We can make artificially make this so by padding shorter sequences with a special symbol not otherwise used (e.g. 0)

the	dog	ran		
he	ran	to	the	house
he	stopped			
he	went	inside		

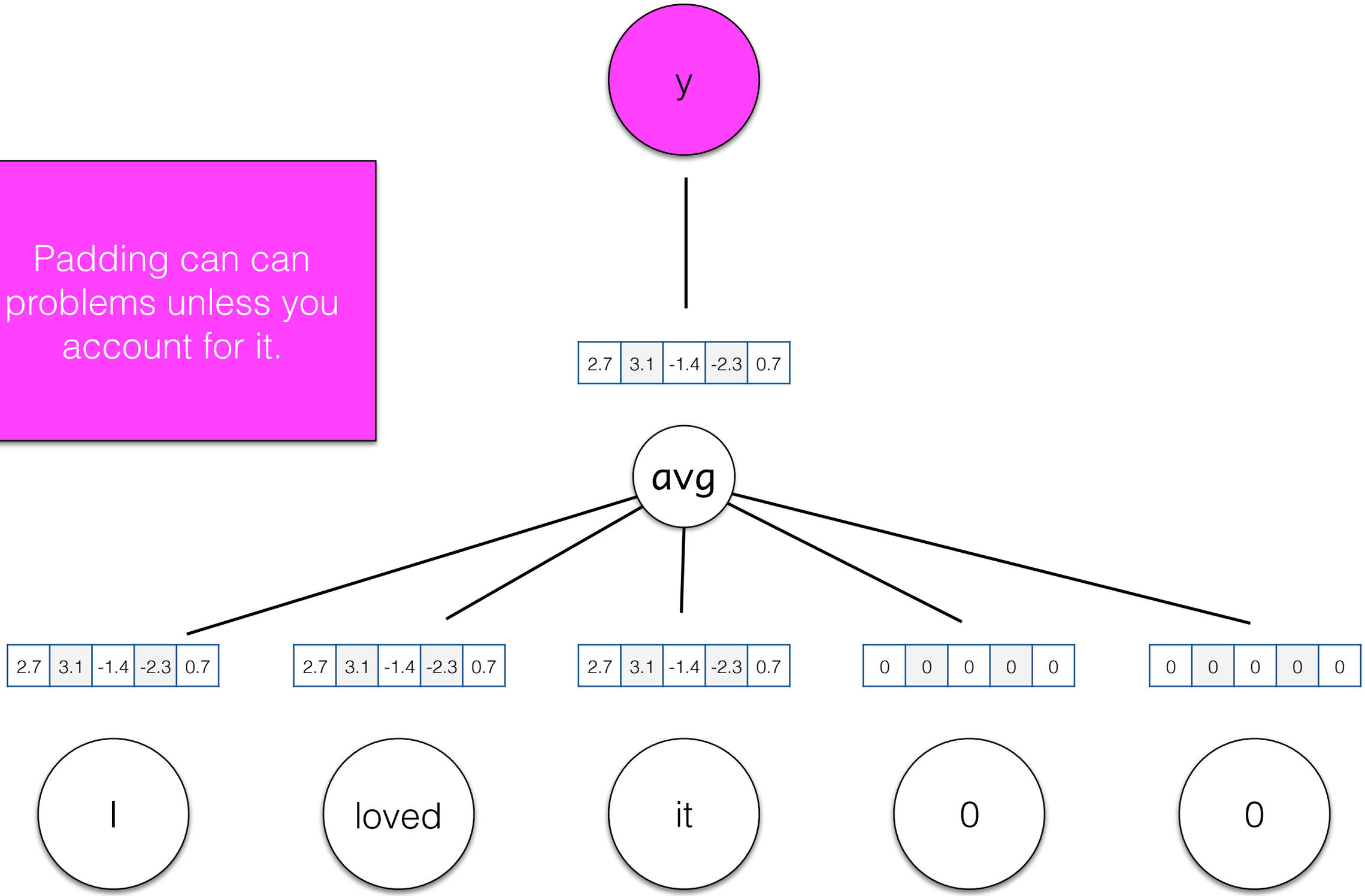
1	3	4		
2	4	5	1	6
2	7			
2	8	9		

word embedding ids

1	3	4	0	0
2	4	5	1	6
2	7	0	0	0
2	8	9	0	0

word embedding ids

Padding can can problems unless you account for it.



Masking

- For sequences that have been padded, ignore all time steps with the padding symbol.