Recurrent neural networks

Lecture 12, 10/24/17

David Sontag

Massachusetts
Institute of
Technology

Acknowledgement: several slides adapted from Tommi Jaakkola

Course announcements

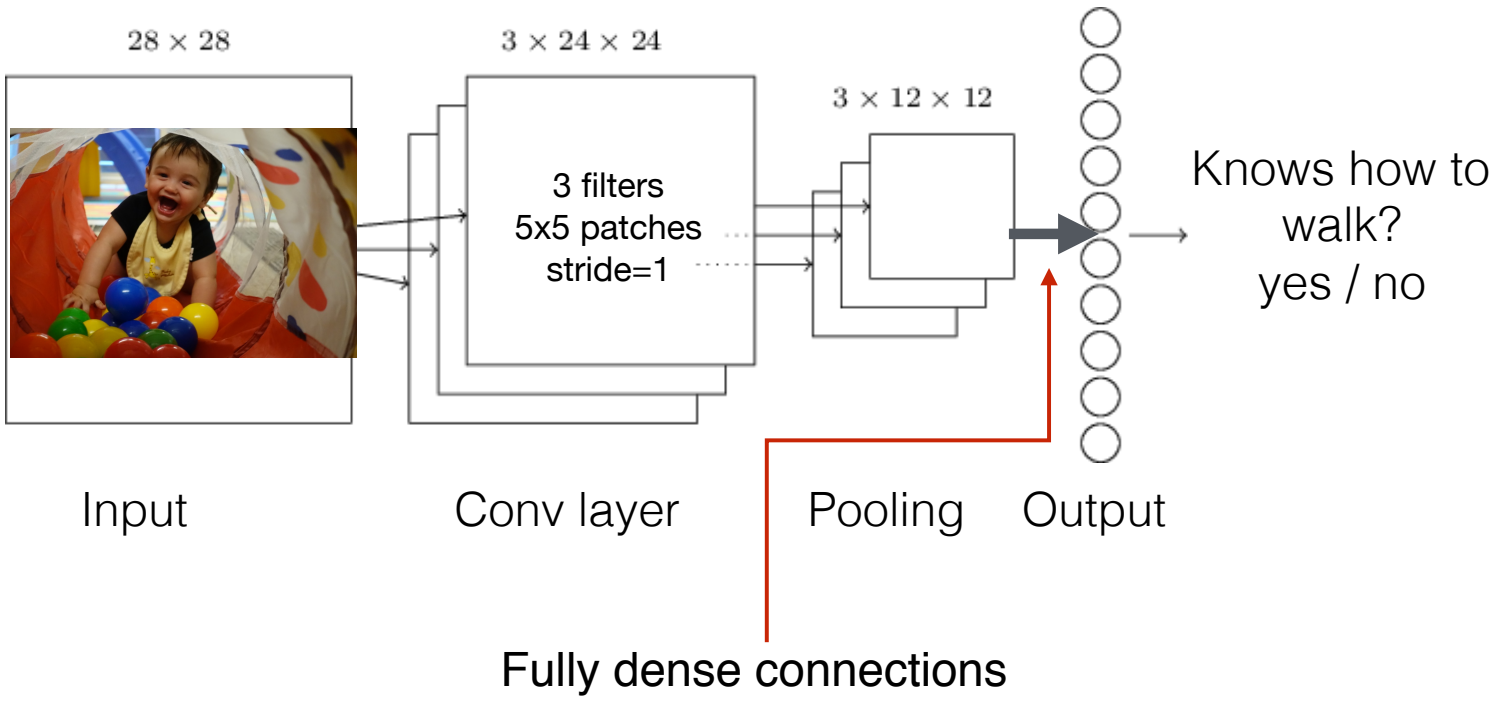
- Exam 1
  - Re-grade requests due Thursday before class
  - Solutions posted Friday or soon after
  - Grade related questions? Discuss with Professor Shah Monday, October 30 in 32-D670, 12:00-12:30pm
- HW2 due today at 11:59pm
- Project Milestone 3 due Thursday at 11:59pm
- Reference for today's class
  - Supervised Sequence Labelling with Recurrent Neural Networks, Alex Graves, 2012. Sections 3.2 and 4
  - **Available for free online**

Today's lecture

- Temporal/sequential data
- Recurrent neural networks
  - Gating, LSTMs
- Language modeling

NN recap (what you know)

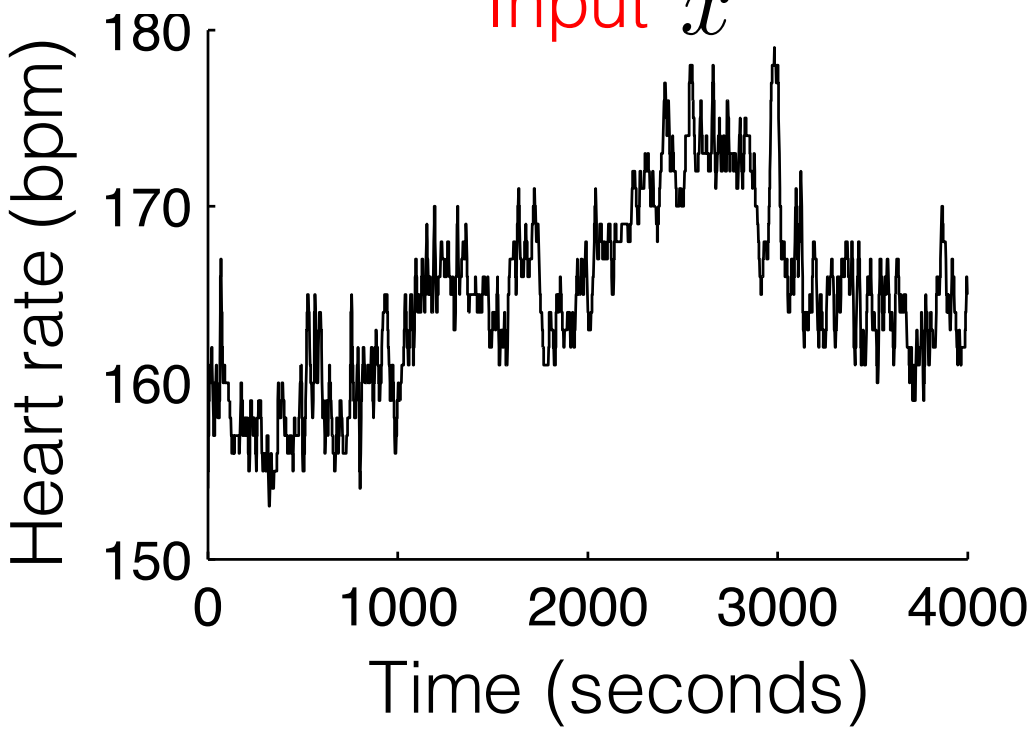
- Feed-forward (layered) neural networks
  - units, weights, activation functions
  - layer-wise forward propagation
  - back-propagation of error
  - stochastic gradient descent
- Convolutional neural networks (CNNs)
  - filters, feature maps, pooling



Temporal/sequence problems

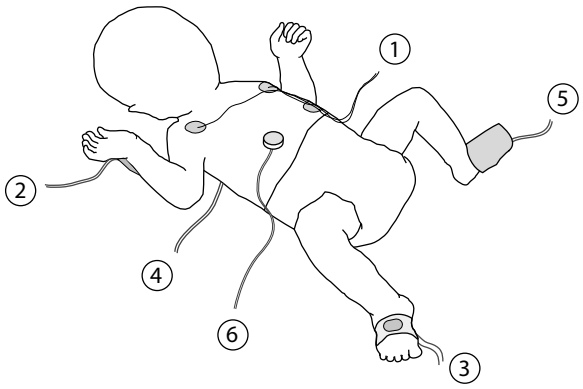
- How to cast as a supervised learning problem?

Input x



Output y

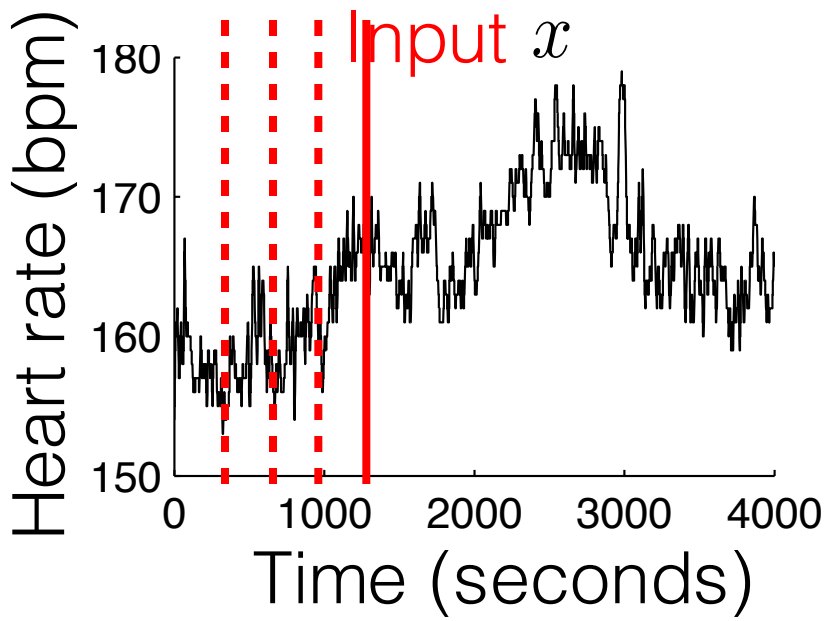
Likelihood of mortality?



Quinn et al., TPAMI 2008

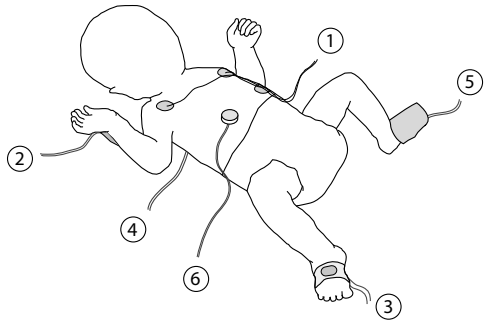
Temporal/sequence problems

- How to cast as a supervised learning problem?



Output y

Likelihood of mortality?

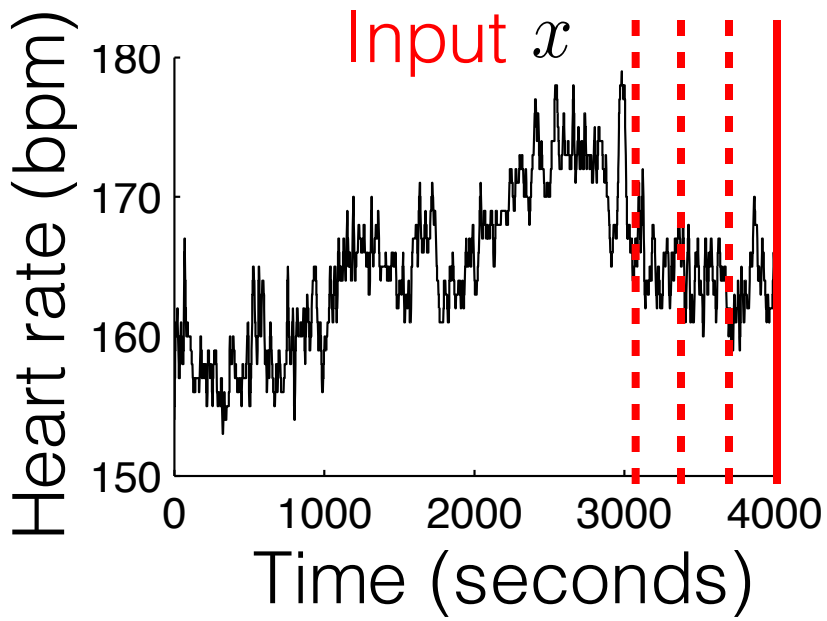


- Historical data can be broken down into feature vectors and target values (sliding window)

$$\underbrace{\langle 155, 160, 165, 167 \rangle}_{x^{(t)}} \qquad \underbrace{0.35}_{y^{(t)}}$$

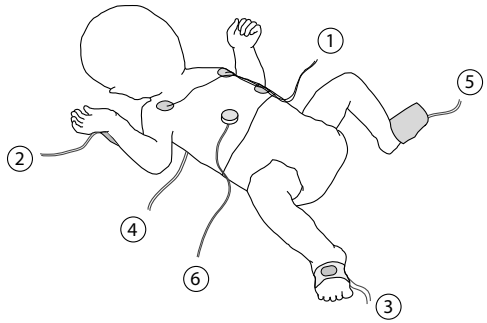
Temporal/sequence problems

- How to cast as a supervised learning problem?



Output y

Likelihood of mortality?



- Historical data can be broken down into feature vectors and target values (sliding window)

$$\underbrace{\langle 170, 160, 155, 160 \rangle}_{x^{(t)}} \qquad \underbrace{0.24}_{y^{(t)}}$$

Temporal/sequence problems

- Language modeling: what comes next?

The weather yesterday was …

Temporal/sequence problems

- Language modeling: what comes next?

The weather yesterday was ...

was $\begin{bmatrix} 0 \\ \vdots \\ 1 \\ 0 \end{bmatrix}$

yesterday $\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$

?

$x^{(t)}$ $y^{(t)}$

Temporal/sequence problems

- Language modeling: what comes next?

The weather yesterday was …

yesterday $\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$

weather $\begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}$

$x^{(t)}$

was

$y^{(t)}$

Called a *trigram model*

Limited history!

Temporal/sequence problems

- Language modeling: what comes next?

The weather yesterday | **was …**

Alternative representation using *bag of words*:

{yesterday, weather, the}

$$\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

was

Loses word order!

$x^{(t)}$ $y^{(t)}$

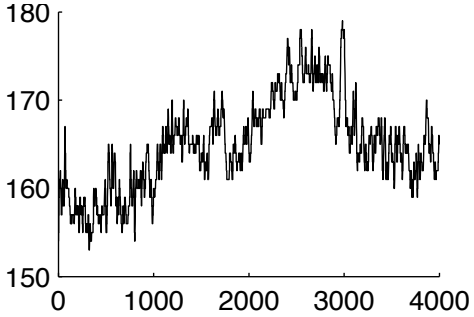
What are we missing?

- Sequence prediction problems can be recast in a form amenable to feed-forward neural networks
- But we have to engineer how "history" is mapped to a vector (representation). This vector is then fed into, e.g., a neural network
  - how many steps back should we look at?
  - how to retain important items mentioned far back?

- Instead, we would like to learn how to encode the "history" into a vector

Learning to encode/decode

- Health care / time-series modeling



Healthy

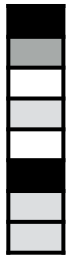
- Sentiment classification

The battery lasted only one hour!

-1

- Machine translation

Como te parece la clase?



How do you like the class?

encoding **decoding**

Key concepts

- **Encoding**
  - e.g., mapping a sequence to a vector
- **Decoding**
  - e.g., mapping a vector to, e.g., a sequence

Encoding everything

words $\begin{bmatrix} .1 \\ .3 \\ .4 \end{bmatrix} \begin{bmatrix} .7 \\ .1 \\ .0 \end{bmatrix} \begin{bmatrix} .2 \\ .8 \\ .3 \end{bmatrix}$ ...

“Efforts and courage are not enough without purpose and direction” — JFK

sentences $\begin{bmatrix} .2 \\ .3 \\ .6 \end{bmatrix}$

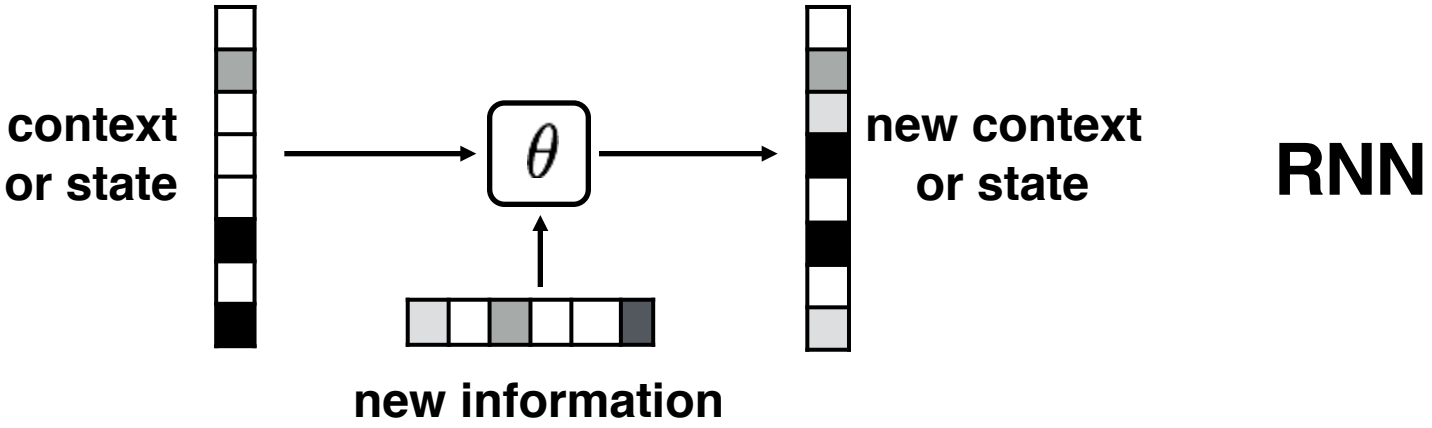
images $\begin{bmatrix} .3 \\ .3 \\ .5 \end{bmatrix}$

$\begin{bmatrix} .2 \\ .4 \\ .6 \end{bmatrix}$

events

Example: encoding sentences

- Easy to introduce adjustable "lego pieces" and optimize them for end-to-end performance

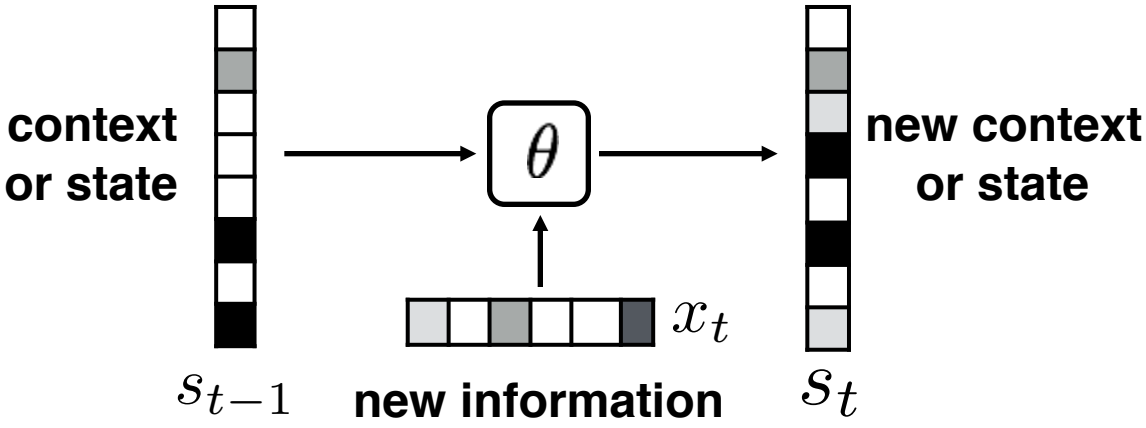


<null>

Efforts and courage are not ...

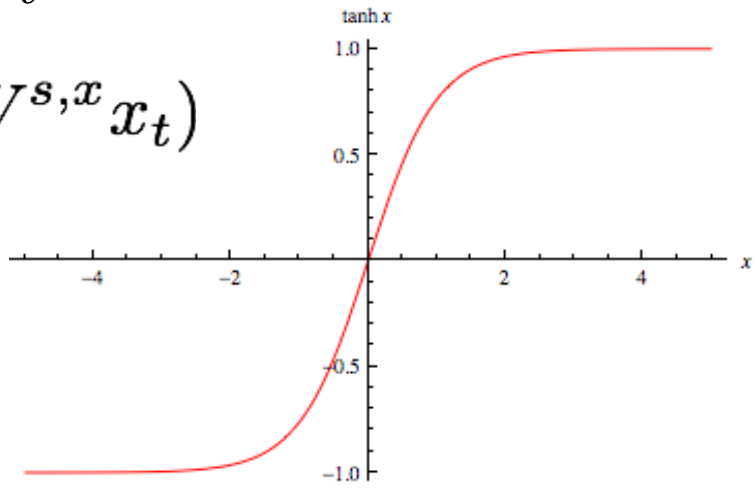
Example: encoding sentences

- Easy to introduce adjustable "lego pieces" and optimize them for end-to-end performance



RNN

$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$

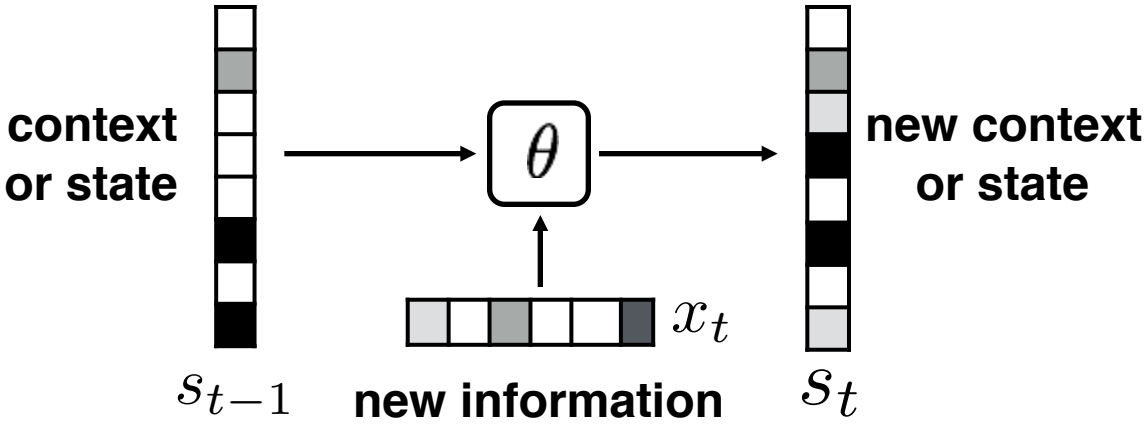


<null>

Efforts and courage are not ...

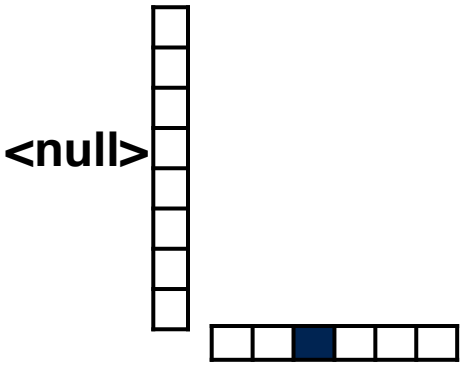
Example: encoding sentences

- Easy to introduce adjustable "lego pieces" and optimize them for end-to-end performance



RNN

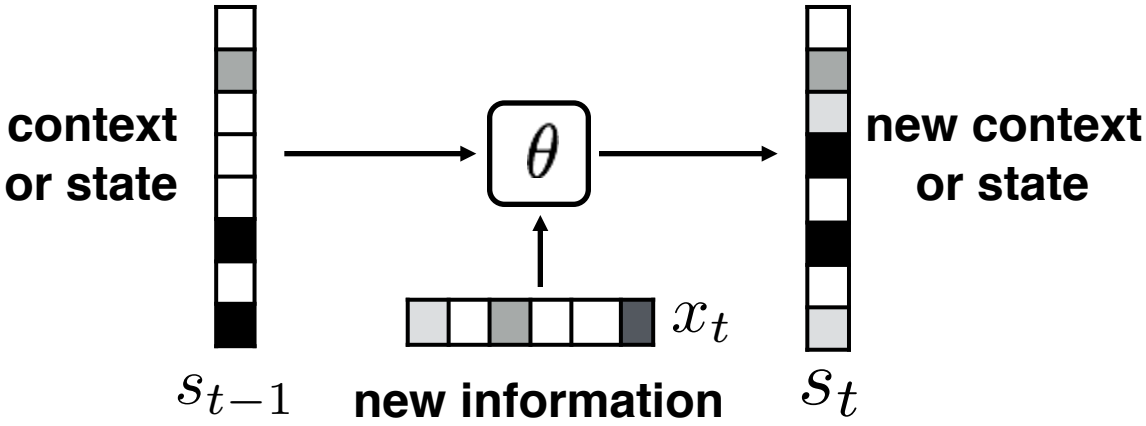
$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$



Efforts and courage are not ...

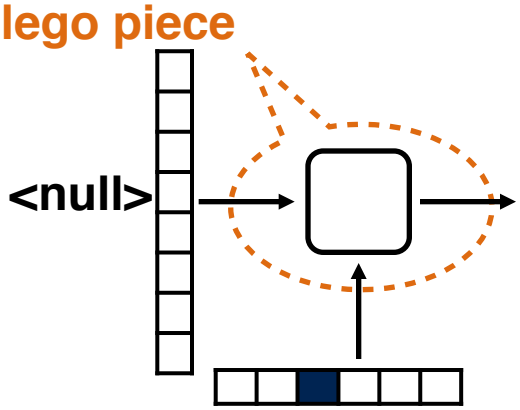
Example: encoding sentences

- Easy to introduce adjustable "lego pieces" and optimize them for end-to-end performance



RNN

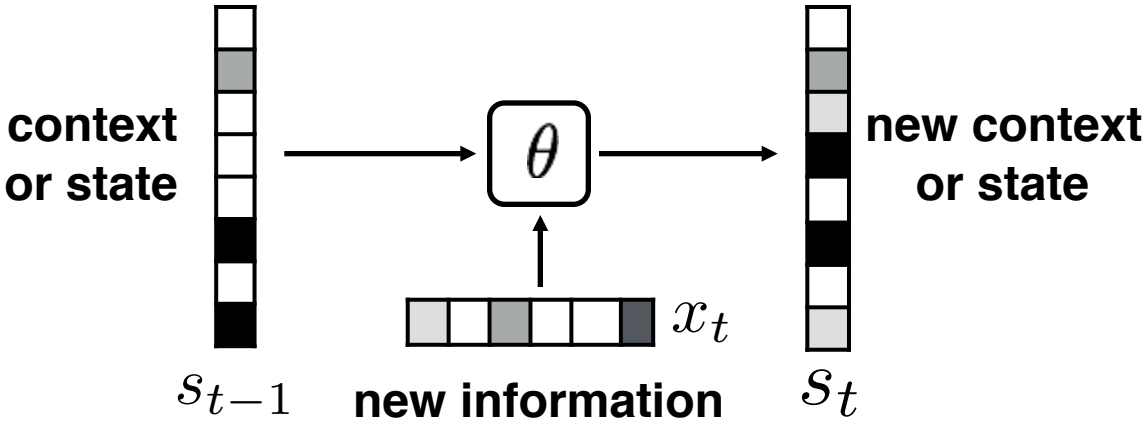
$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$



Efforts and courage are not ...

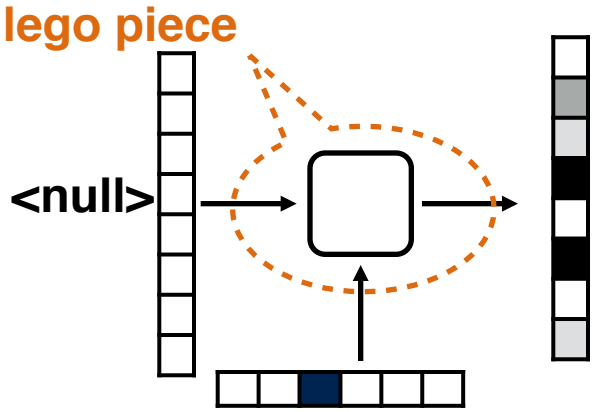
Example: encoding sentences

- Easy to introduce adjustable "lego pieces" and optimize them for end-to-end performance



RNN

$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$

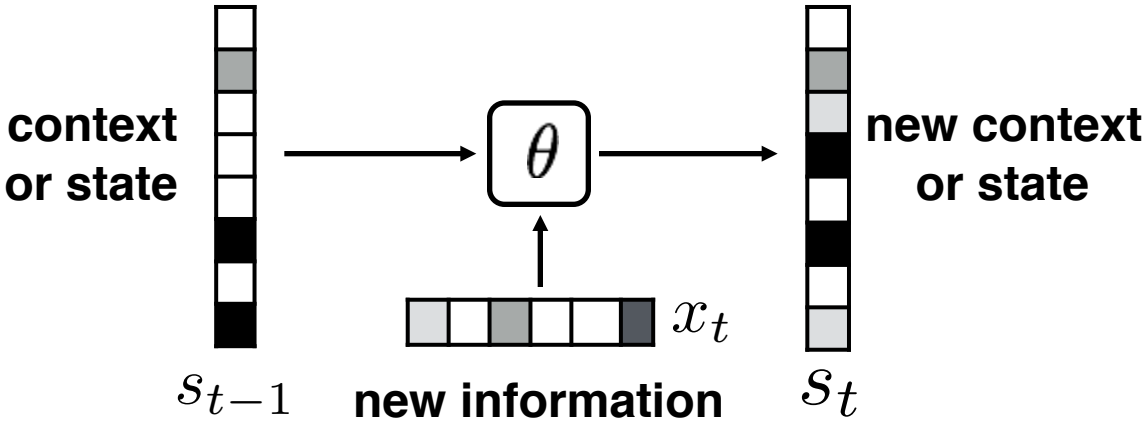


summary of "Efforts"

Efforts and courage are not ...

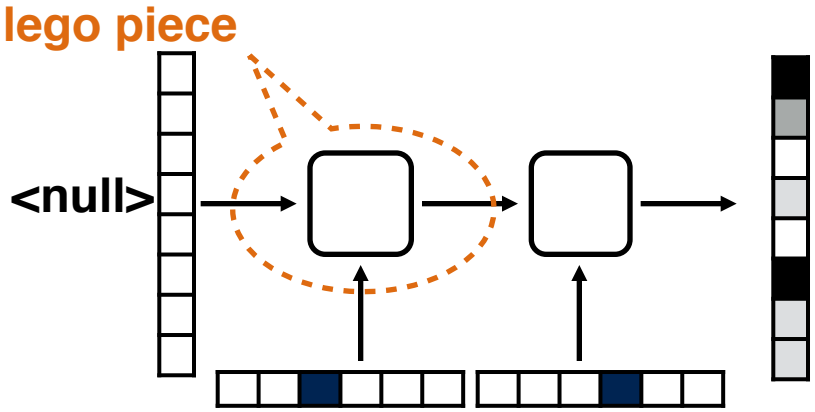
Example: encoding sentences

- Easy to introduce adjustable "lego pieces" and optimize them for end-to-end performance



RNN

$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$

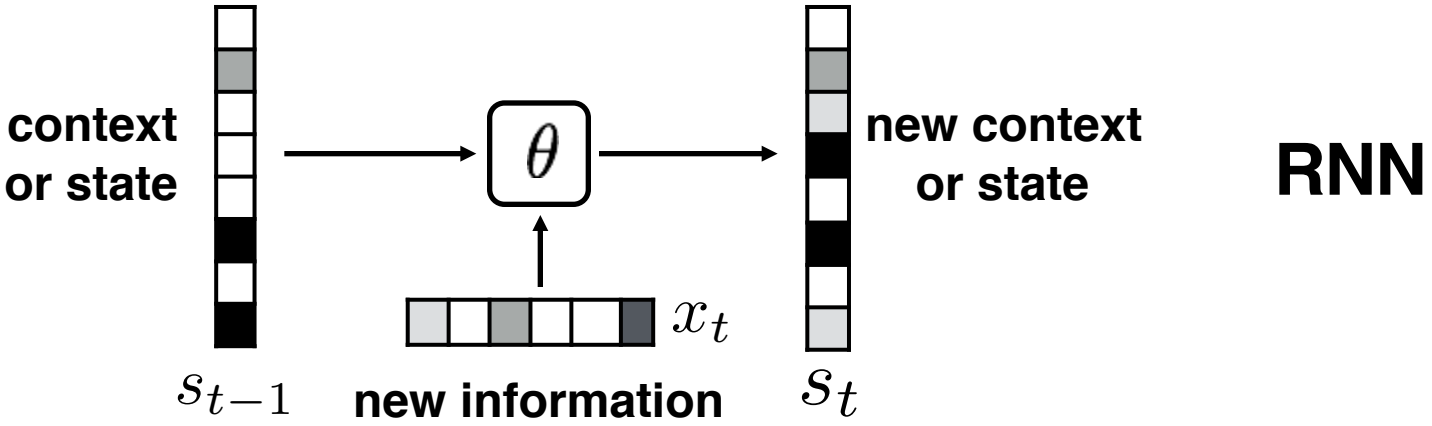


summary of "Efforts and"

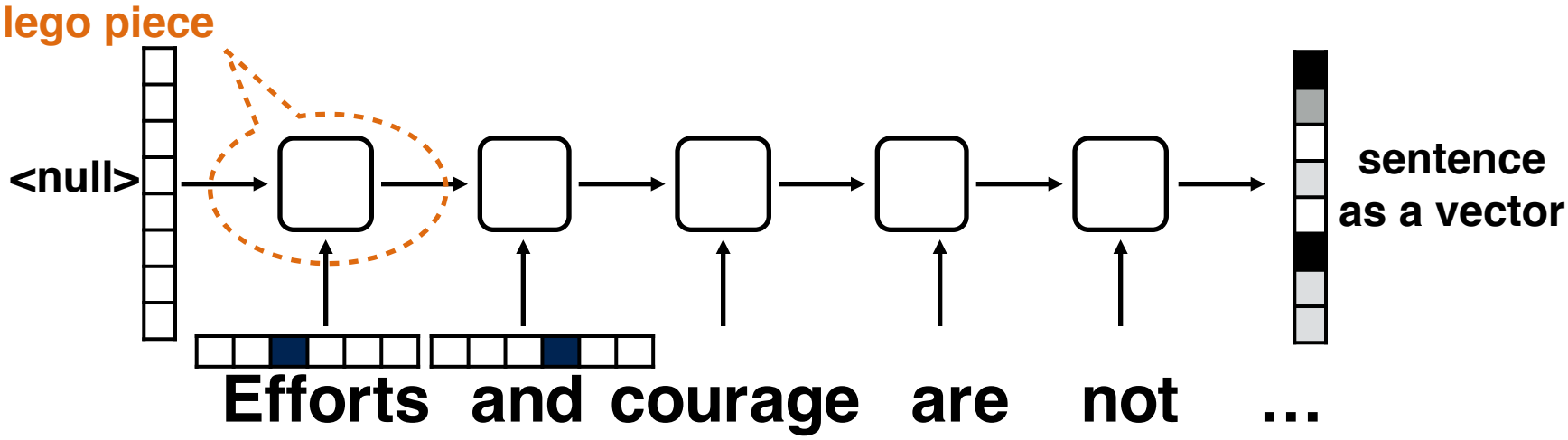
Efforts and courage are not ...

Example: encoding sentences

- Easy to introduce adjustable "lego pieces" and optimize them for end-to-end performance

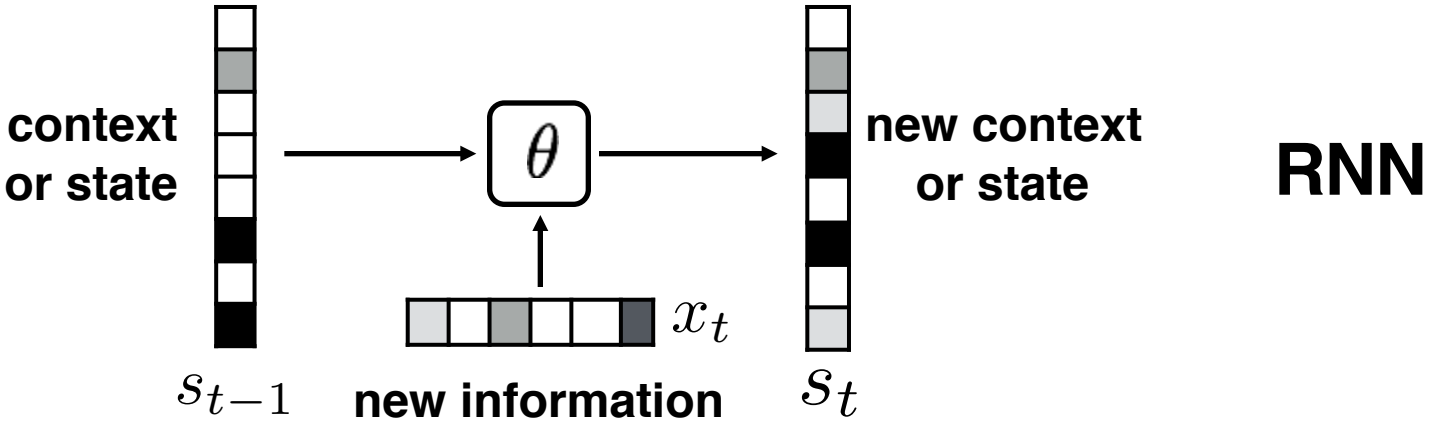


$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$

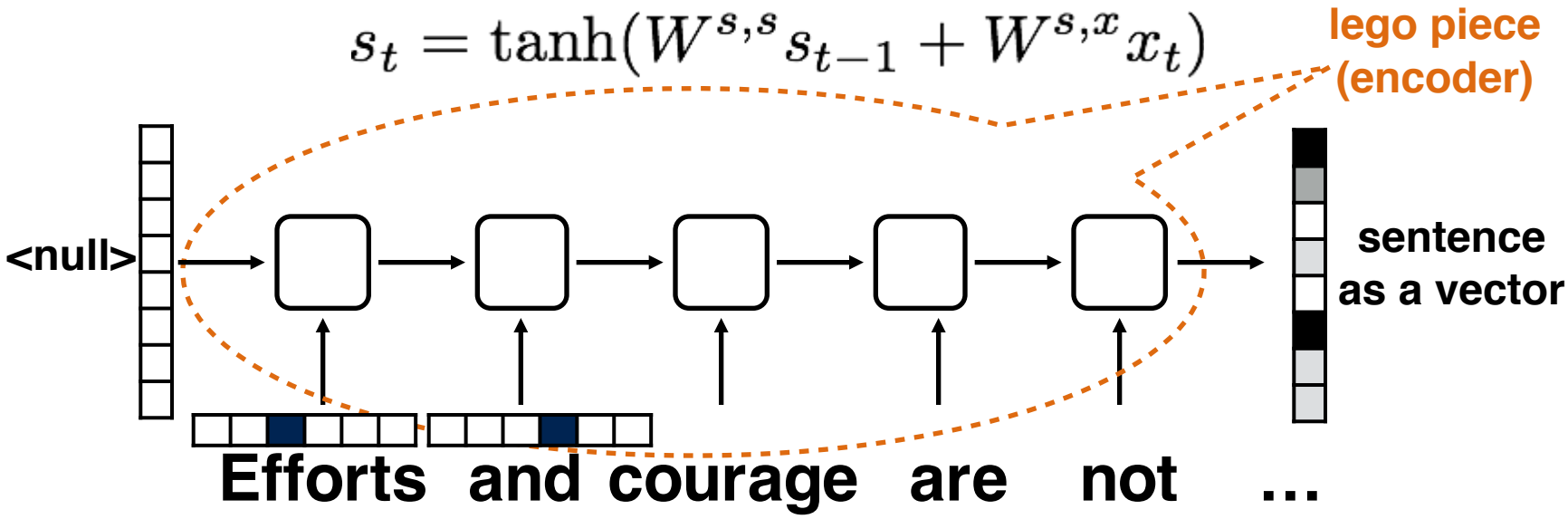


Example: encoding sentences

- Easy to introduce adjustable "lego pieces" and optimize them for end-to-end performance

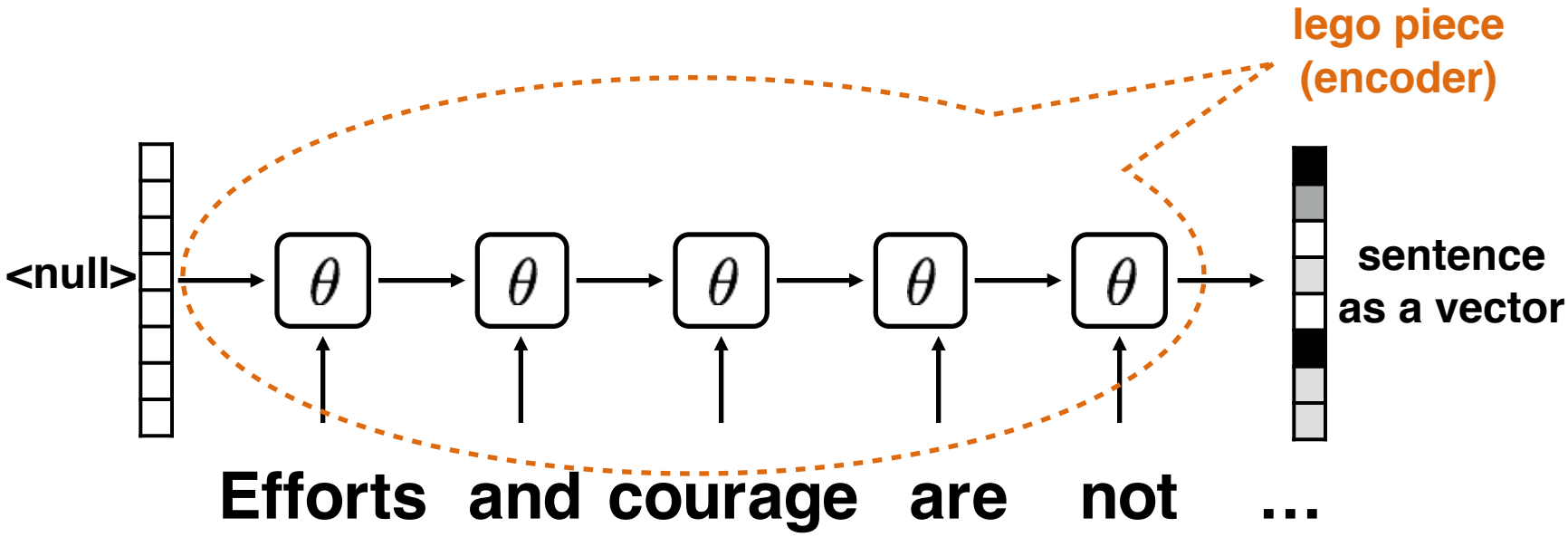


$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$



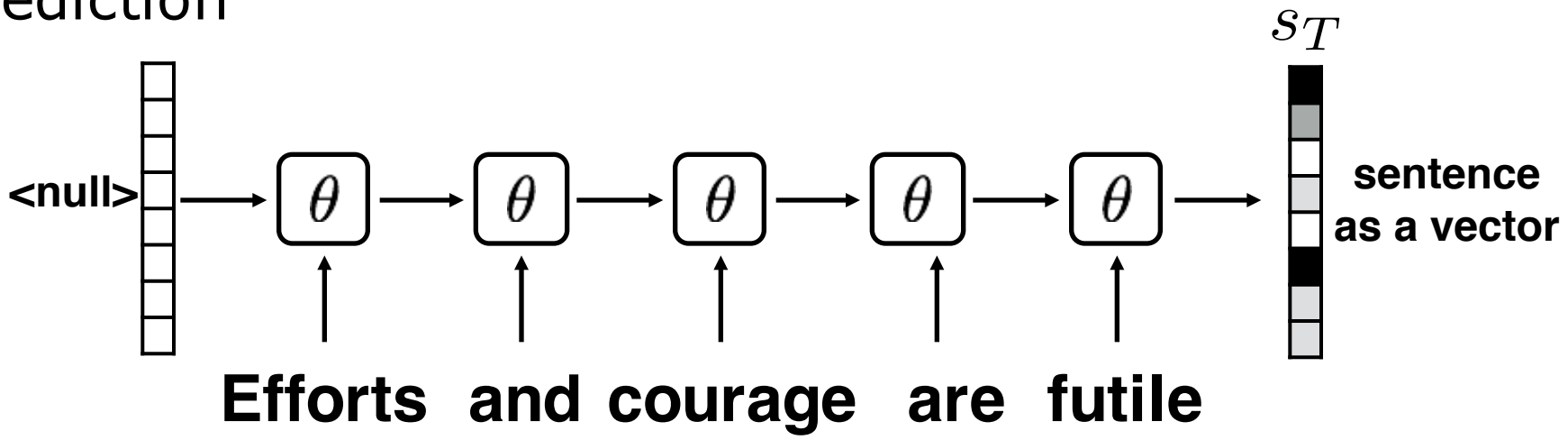
Example: encoding sentences

- There are three differences between the encoder (unfolded RNN) and a standard feed-forward architecture
  - input is received at each layer (per word), not just at the beginning as in a typical feed-forward network
  - the number of layers varies, and depends on the length of the sentence
  - parameters of each layer (representing an application of an RNN) are shared (same RNN at each step)



Example: decoding for prediction

- Simplest example: use the last hidden state for prediction



- E.g., using a logistic or softmax

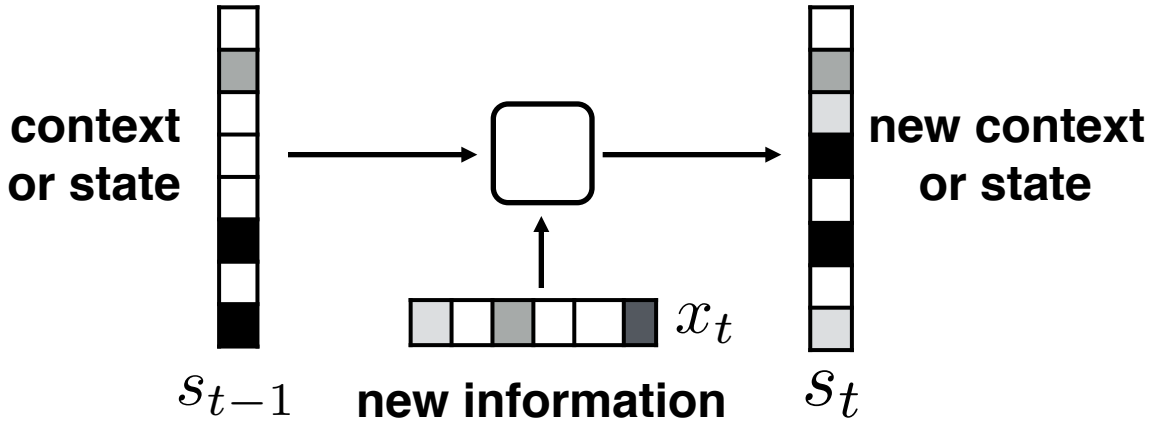
$$\hat{y} \leftarrow \sigma(w \cdot s_T)$$

- Learning (with log-loss) then corresponds to

$$\min_{\theta,w} -\sum_{i=1}^{N}\sum_{k} y_{i,k} \log \hat{y}_{i,k}$$

What's in the box?

- We can make the RNN more sophisticated…

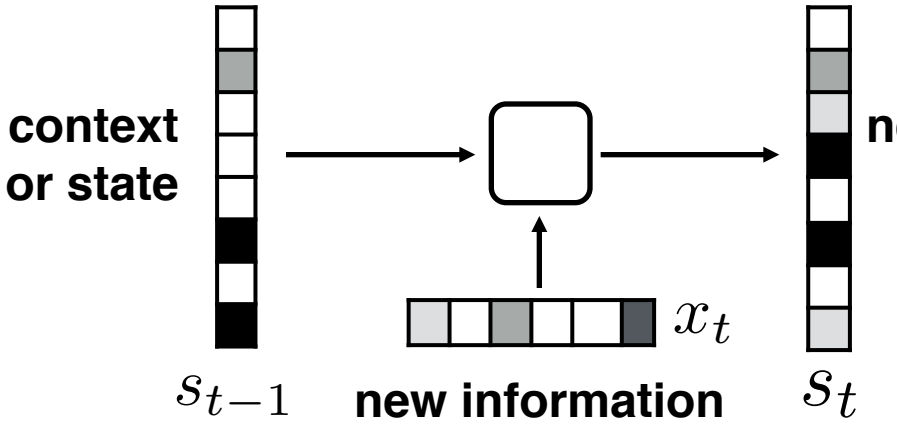


basic RNN

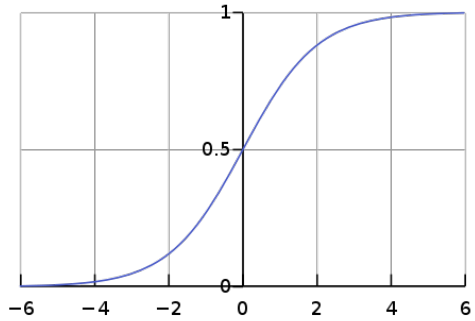
$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$

What's in the box?

- We can make the RNN more sophisticated…



simple gated RNN



Gate vector

$$g_t = \text{sigmoid}(W^{g,s} s_{t-1} + W^{g,x} x_t)$$

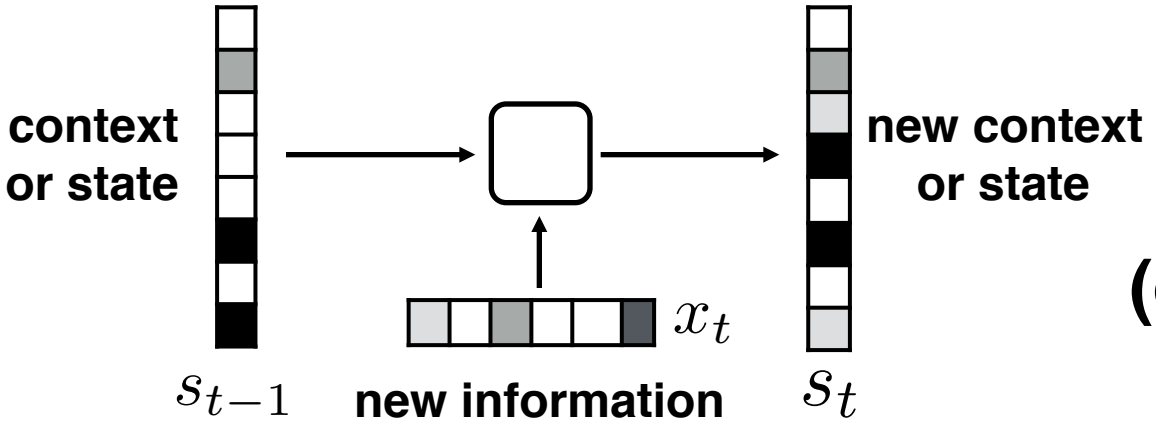
$$s_t = (1 - g_t) \odot s_{t-1} + g_t \odot \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$$

Element-wise multiplication

- Gates (one per dimension) determine whether to keep *past* value or *replace* with new

What's in the box?

- We can make the RNN more sophisticated…



LSTM (one of many)

$$f_t = \text{sigmoid}(W^{f,h} h_{t-1} + W^{f,x} x_t)$$ forget gate

$$i_t = \text{sigmoid}(W^{i,h} h_{t-1} + W^{i,x} x_t)$$ input gate

$$o_t = \text{sigmoid}(W^{o,h} h_{t-1} + W^{o,x} x_t)$$ output gate

s_t

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W^{c,h} h_{t-1} + W^{c,x} x_t)$$ memory cell

$$h_t = o_t \odot \tanh(c_t)$$ visible state

What's in the box?

- Long short-term memory (LSTM)

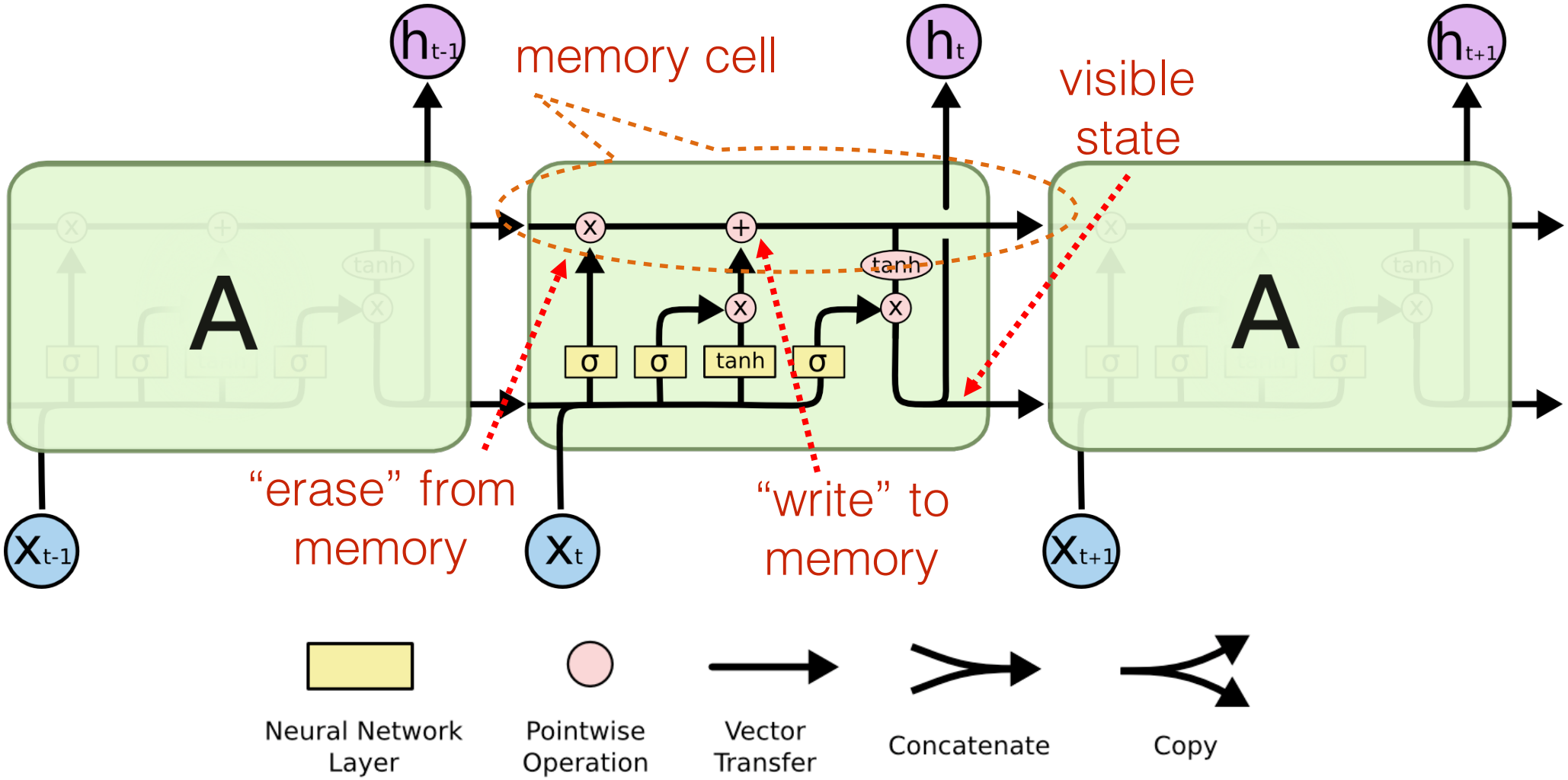


Figure: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Language modeling

- RNNs can be used to model the likelihood of a sequence:

$$\Pr(x_1, x_2, \ldots, x_T) = \Pr(x_1) \prod_{t=2}^{T} \Pr(x_t \mid x_1, \ldots x_{t-1})$$

- Key idea: use RNN hidden state to summarize first *t-1* inputs
- Instead of predicting (outputting) only *at end* of sequence, we output at every time step
- This provides significantly more supervision for training (a form of unsupervised learning)

- Gives a *generative model* of language, music, …

Generating baby names

- Input: 8000 baby names
- One character per time-step
- Learn a RNN to maximize the likelihood of the sequences
- Then, sample from the distribution!

- *Rudi Levette Berice Lussa Hany Mareanne Chrestina Carissy Marylen Hammine Janye Marlise Jacacrie Hendred Romand Charienna Nenotto Ette Dorane Wallen Marly Darine Salina Elvyn Ersia Maralena Minoria Ellia Charmin Antley Nerille Chelon Walmor Evena Jeryly Stachon Charisa Allisa Anatha Cathanie Geetra Alexie Jerin Cassen Herbett Cossie Velen Daurenge Robester Shermond Terisa Licia Roselen Ferine Jayn Lusine Charyanne Sales Sanny Resa Wallon Martine Merus Jelen Candica Wallin Tel Rachene Tarine Ozila Ketia Shanne Arnande Karella Roselina Alessia Chasty Deland Berther Geamar Jackein Mellisand Sagdy Nenc Lessie Rasemy Guen Gavi Milea Anneda Margoris Janin Rodelin Zeanna Elyne Janah Ferzina Susta Pey Castina*

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

Generating handwriting

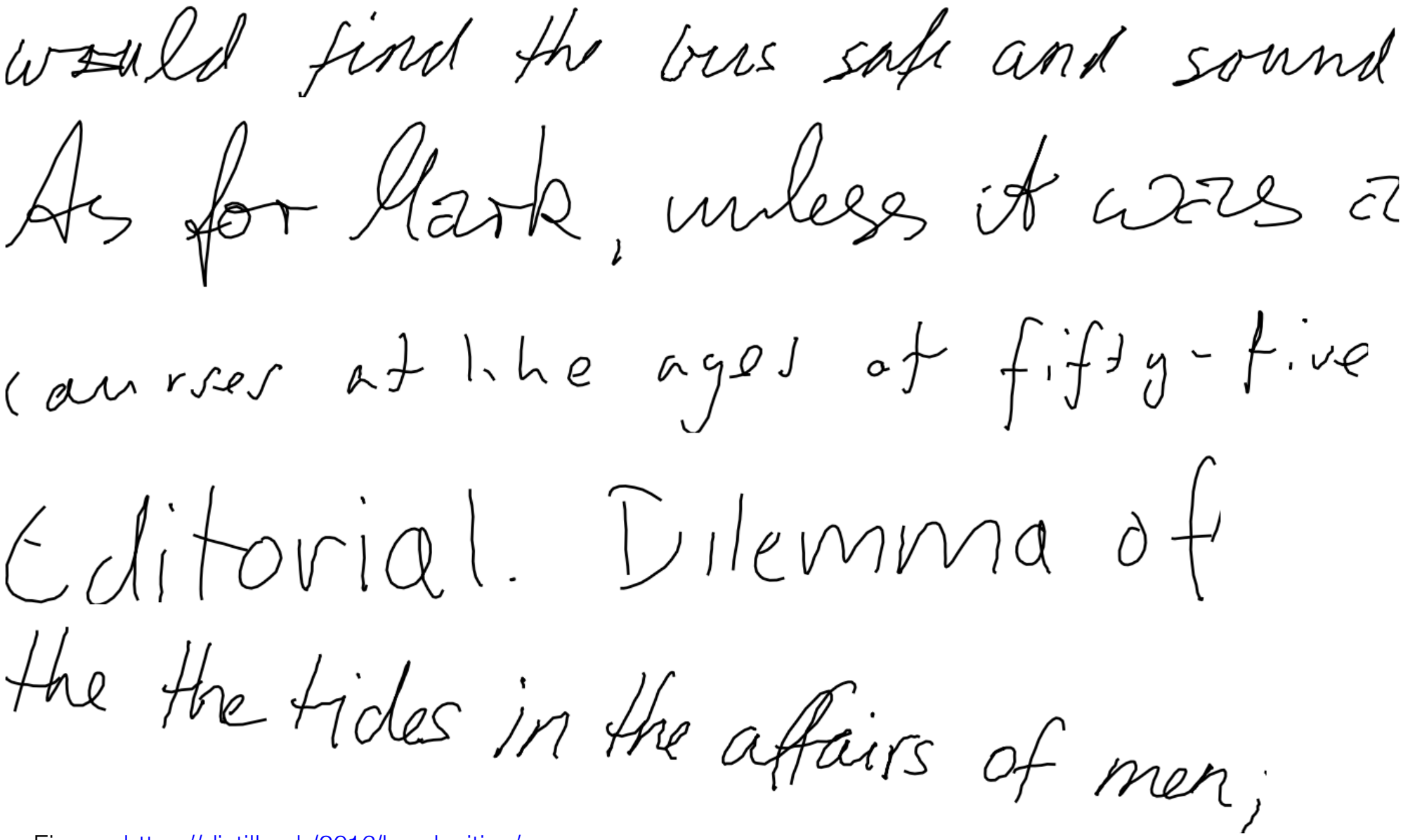
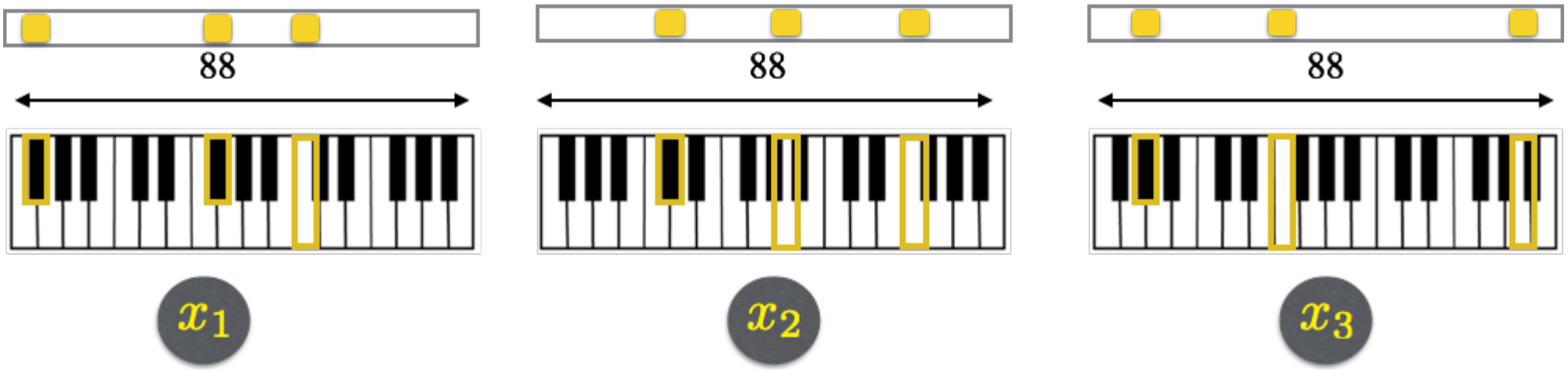


Figure: https://distill.pub/2016/handwriting/

Generating music

- Learn a RNN on J. S. Bach chorales (Lewandowski et al., '12)



Listen:

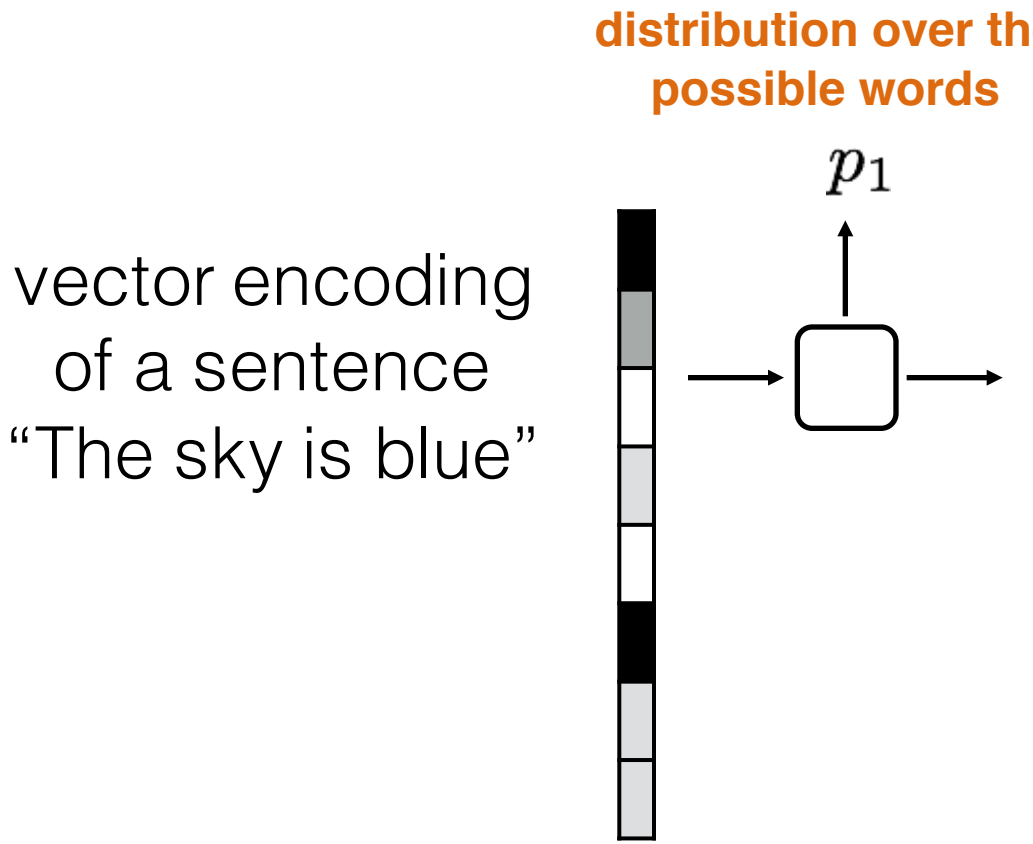
Decoding (into a sentence)

- Our RNN now needs to also produce an output (e.g., a word) as well as update its state

vector encoding
of a sentence
“The sky is blue”

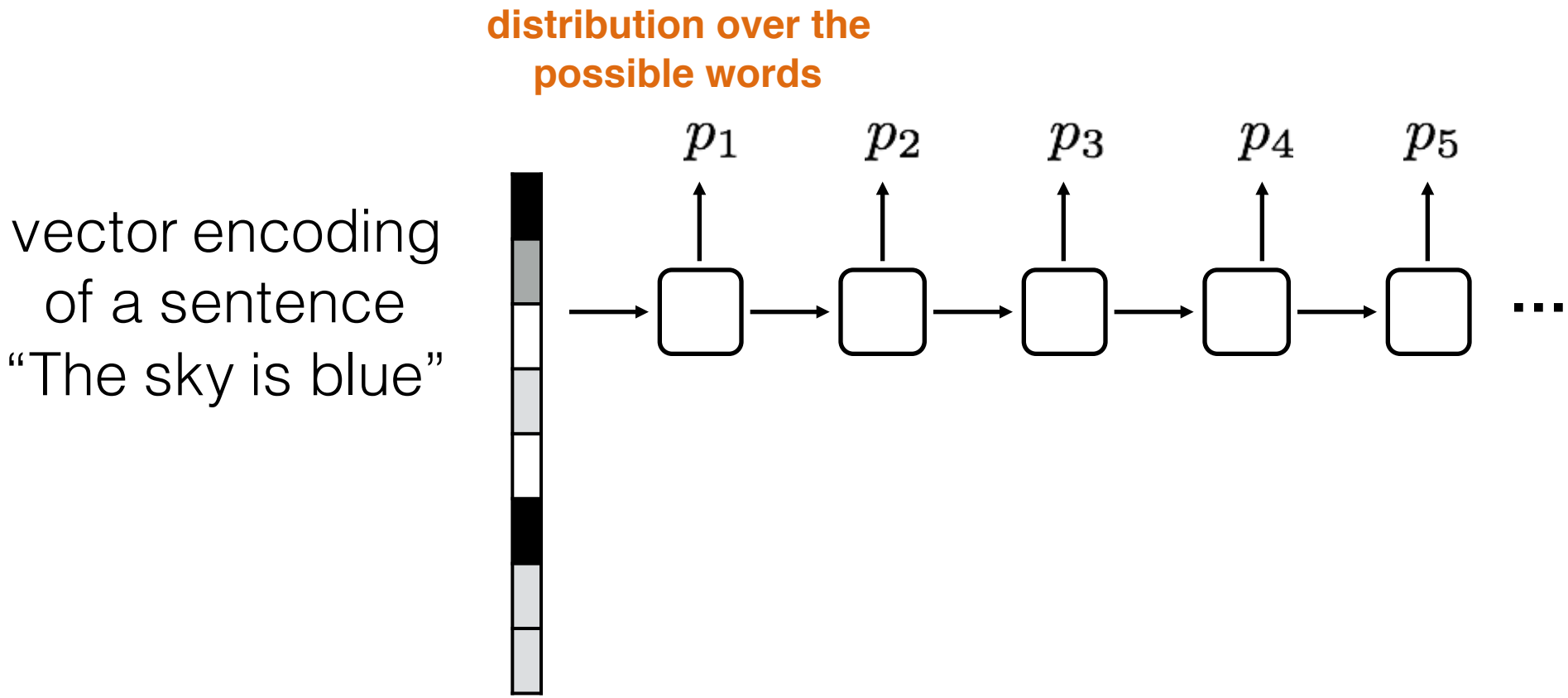
Decoding (into a sentence)

- Our RNN now needs to also produce an output (e.g., a word) as well as update its state



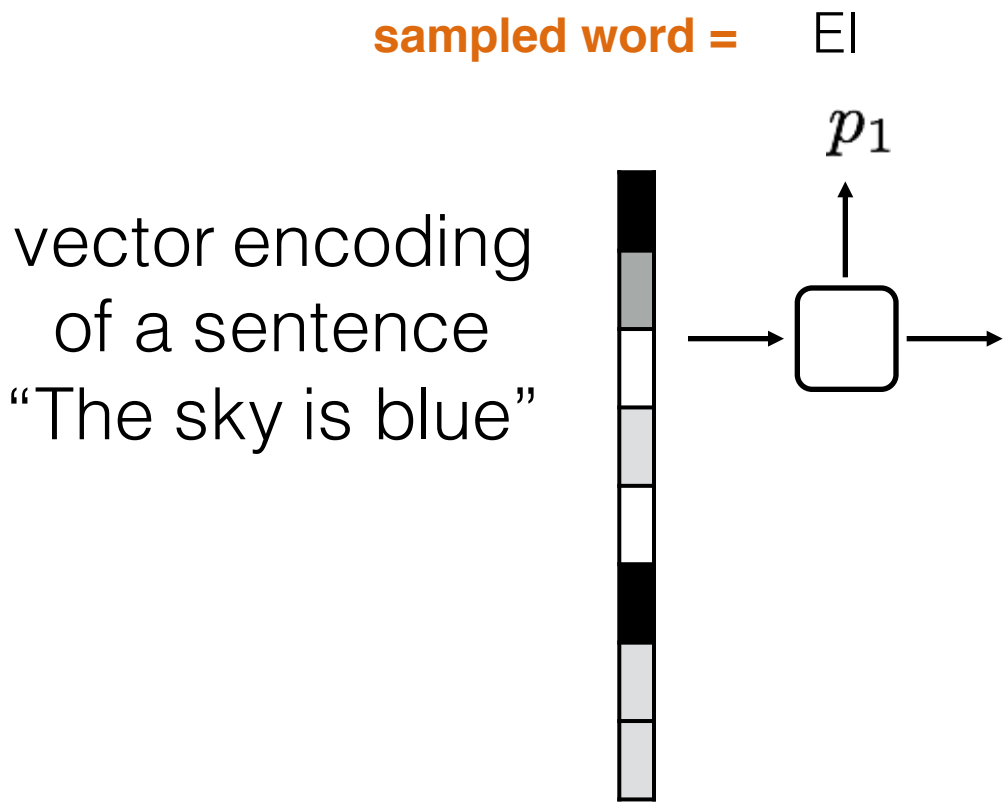
Decoding (into a sentence)

- Our RNN now needs to also produce an output (e.g., a word) as well as update its state



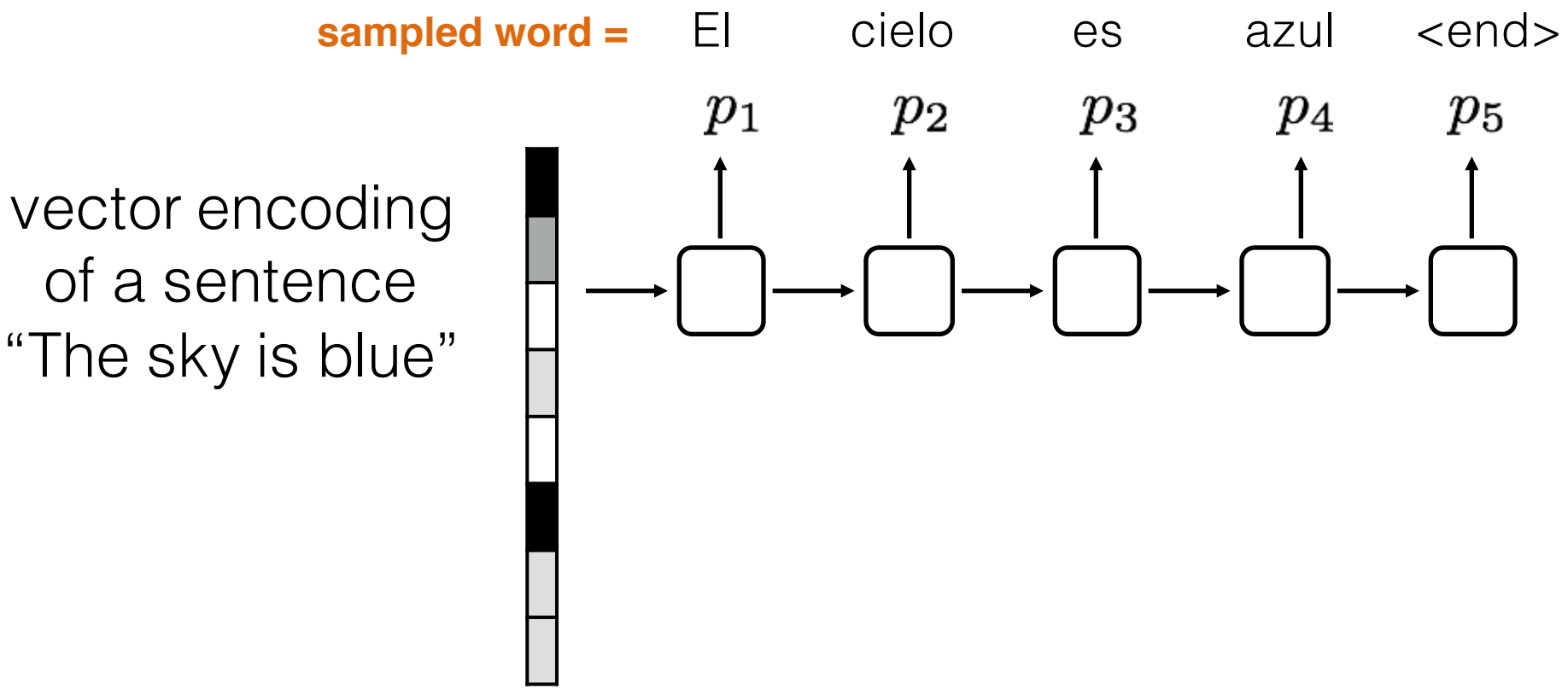
Decoding (into a sentence)

- Our RNN now needs to also produce an output (e.g., a word) as well as update its state



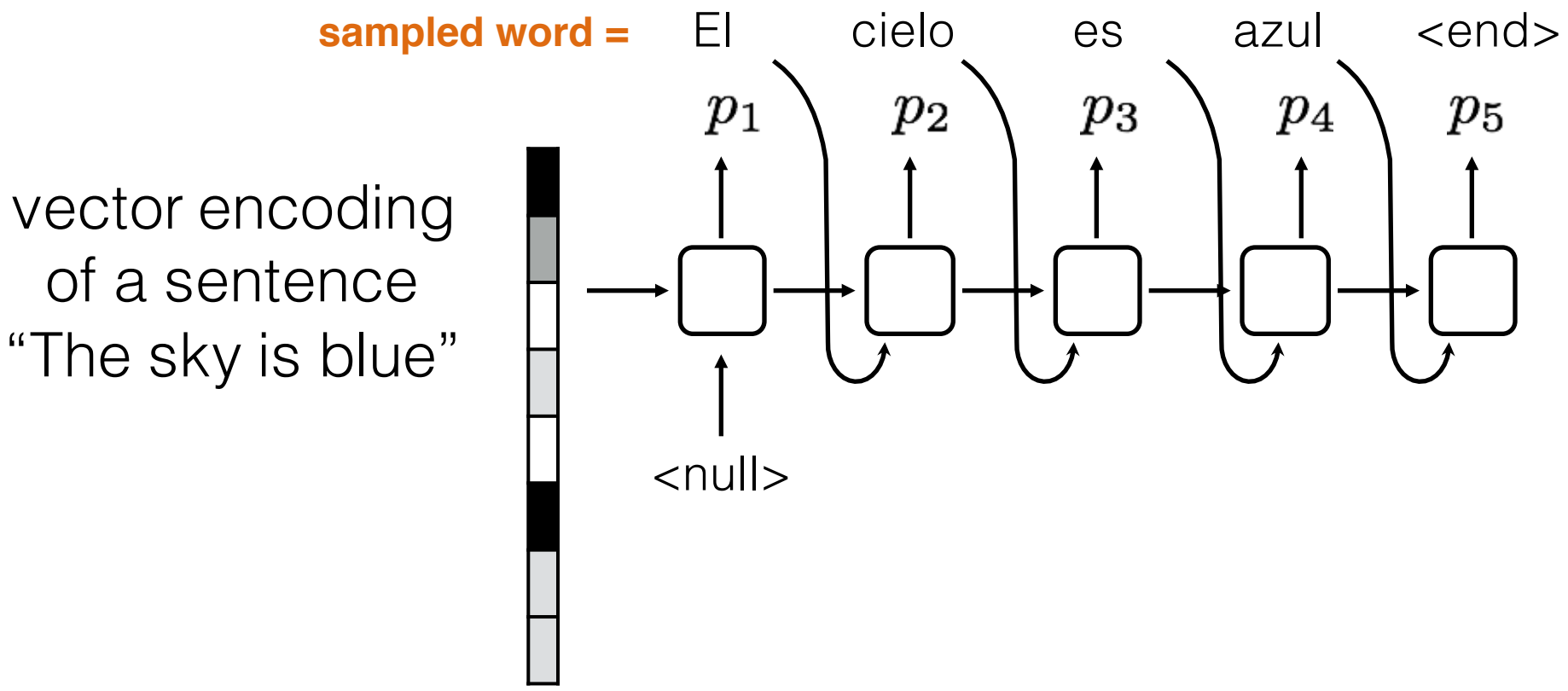
Decoding (into a sentence)

- Our RNN now needs to also produce an output (e.g., a word) as well as update its state



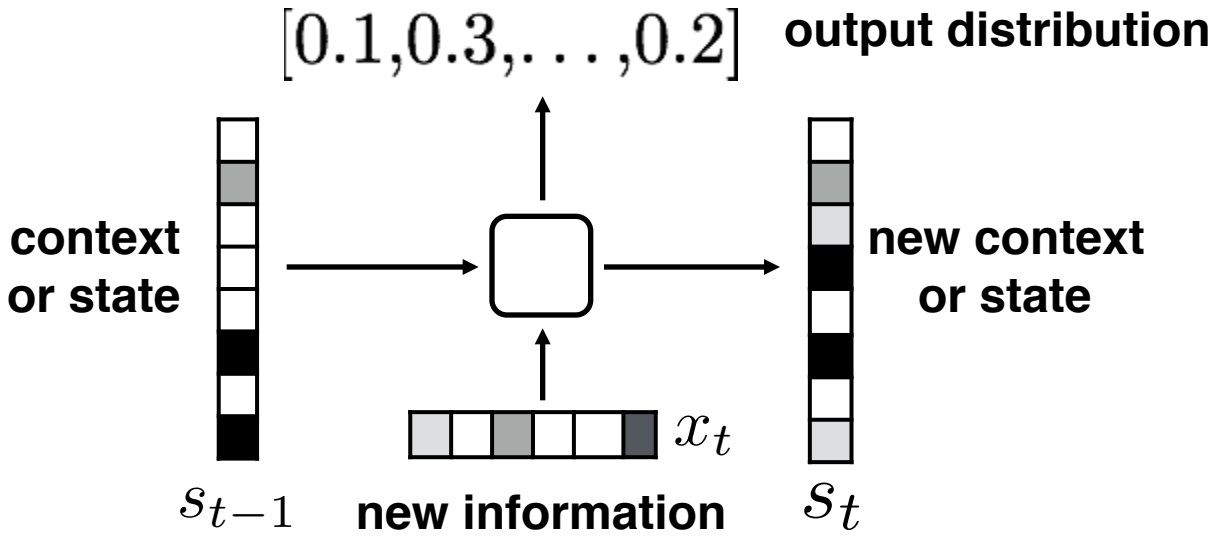
Decoding (into a sentence)

- Our RNN now needs to also produce an output (e.g., a word) as well as update its state
- The output is fed in as an input (to gauge what's left)



Decoding: what's in the box

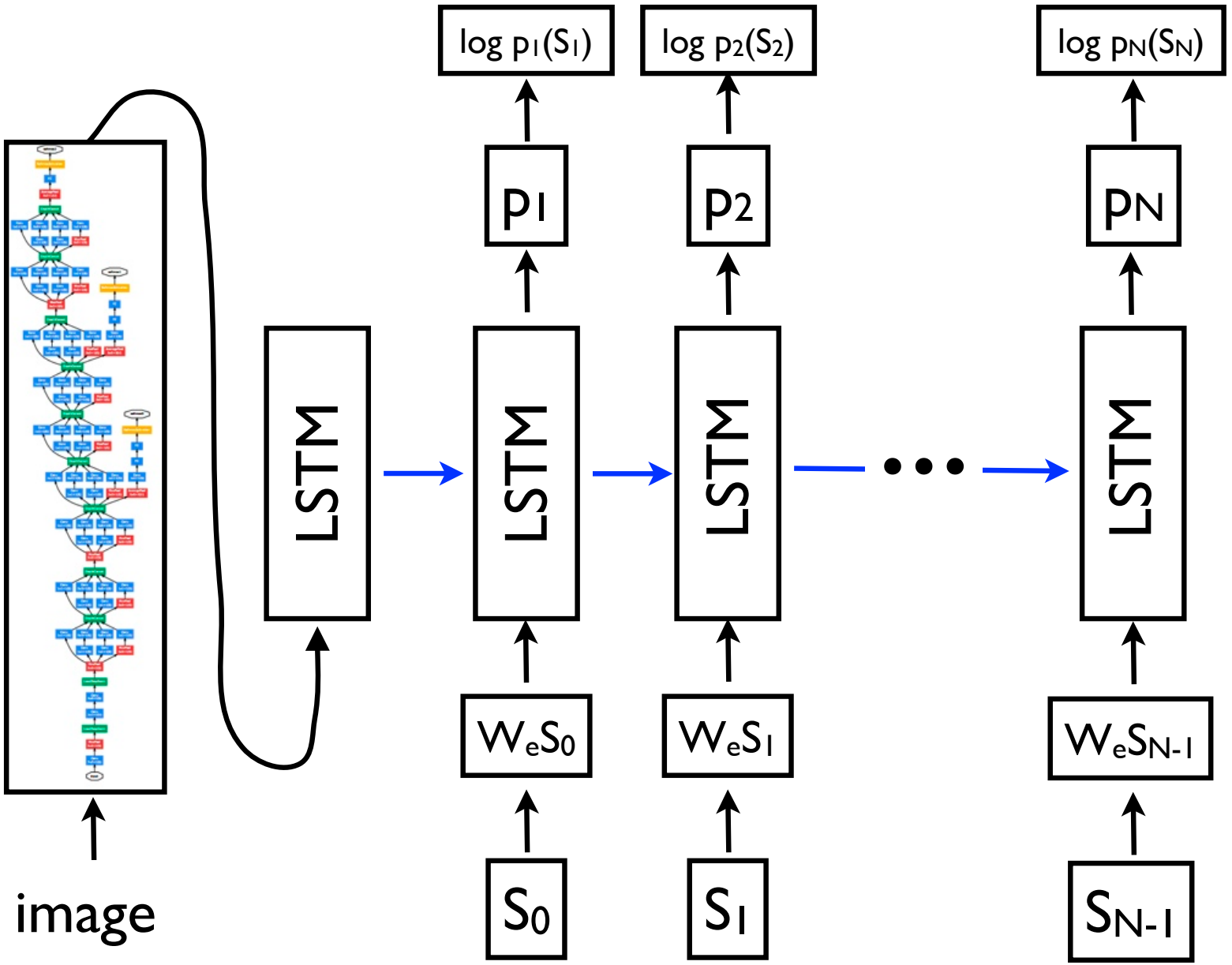
- Our RNN now needs to also produce an output (e.g., a word) as well as update its state



$$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t) \quad \text{state}$$

$$p_t = \text{softmax}(W^o s_t) \quad \text{output distribution}$$

Mapping images to text



Examples

A person riding a motorcycle on a dirt road.

Two dogs play in the grass.

A skateboarder does a trick on a ramp.

A dog is jumping to catch a frisbee.

A group of young people playing a game of frisbee.

Two hockey players are fighting over the puck.

A little girl in a pink hat is blowing bubbles.

A refrigerator filled with lots of food and drinks.

A herd of elephants walking across a dry grass field.

A close up of a cat laying on a couch.

A red motorcycle parked on the side of the road.

A yellow school bus parked in a parking lot.

Describes without errors	Describes with minor errors	Somewhat related to the image	Unrelated to the image

Key things

- Neural networks for sequences
  - encoding/decoding
- RNNs, unfolded
  - state evolution, gates
  - relation to feed-forward neural networks
  - back-propagation (conceptually)
- Issues: vanishing/exploding gradient
- LSTM (operationally)