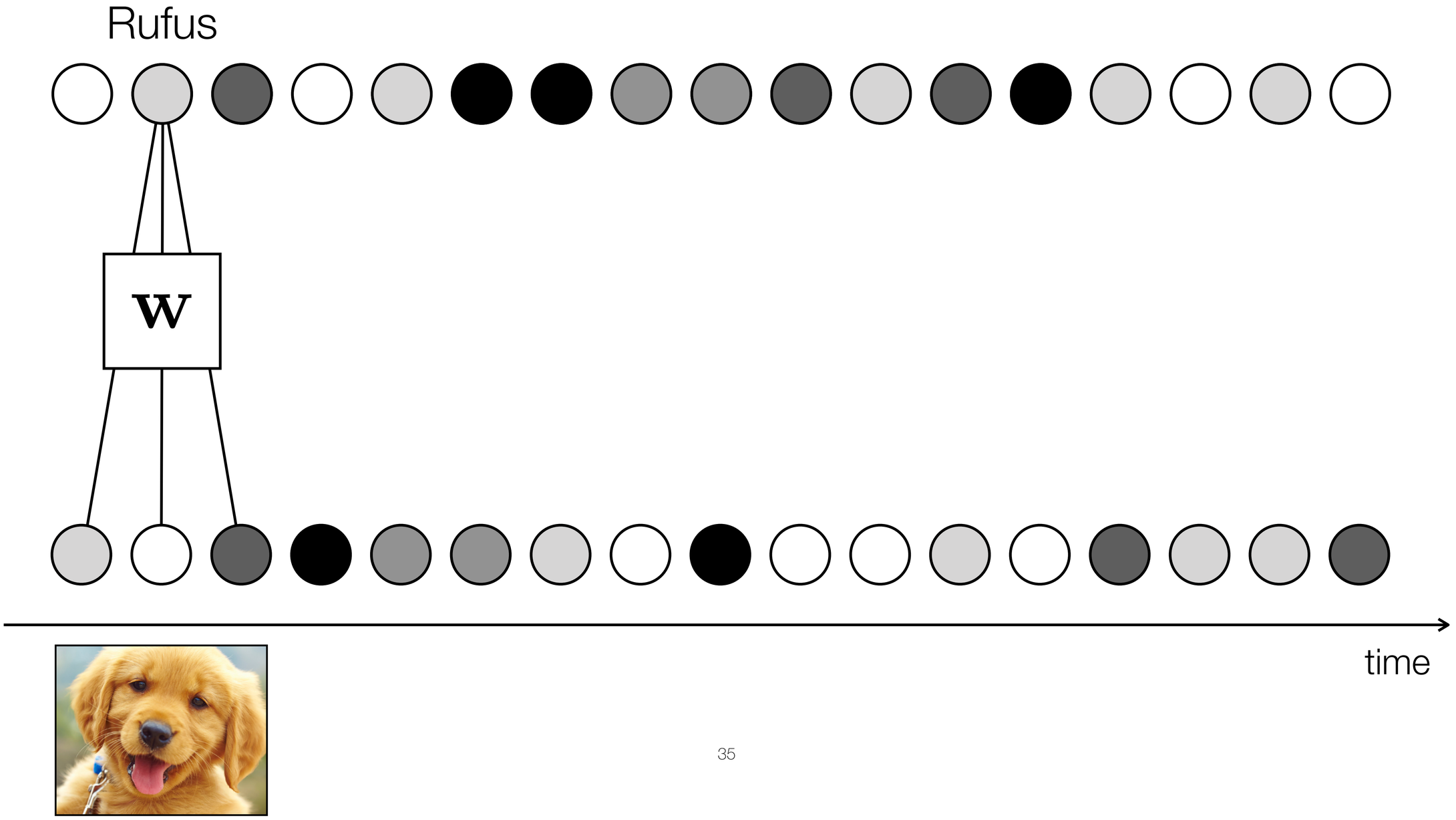
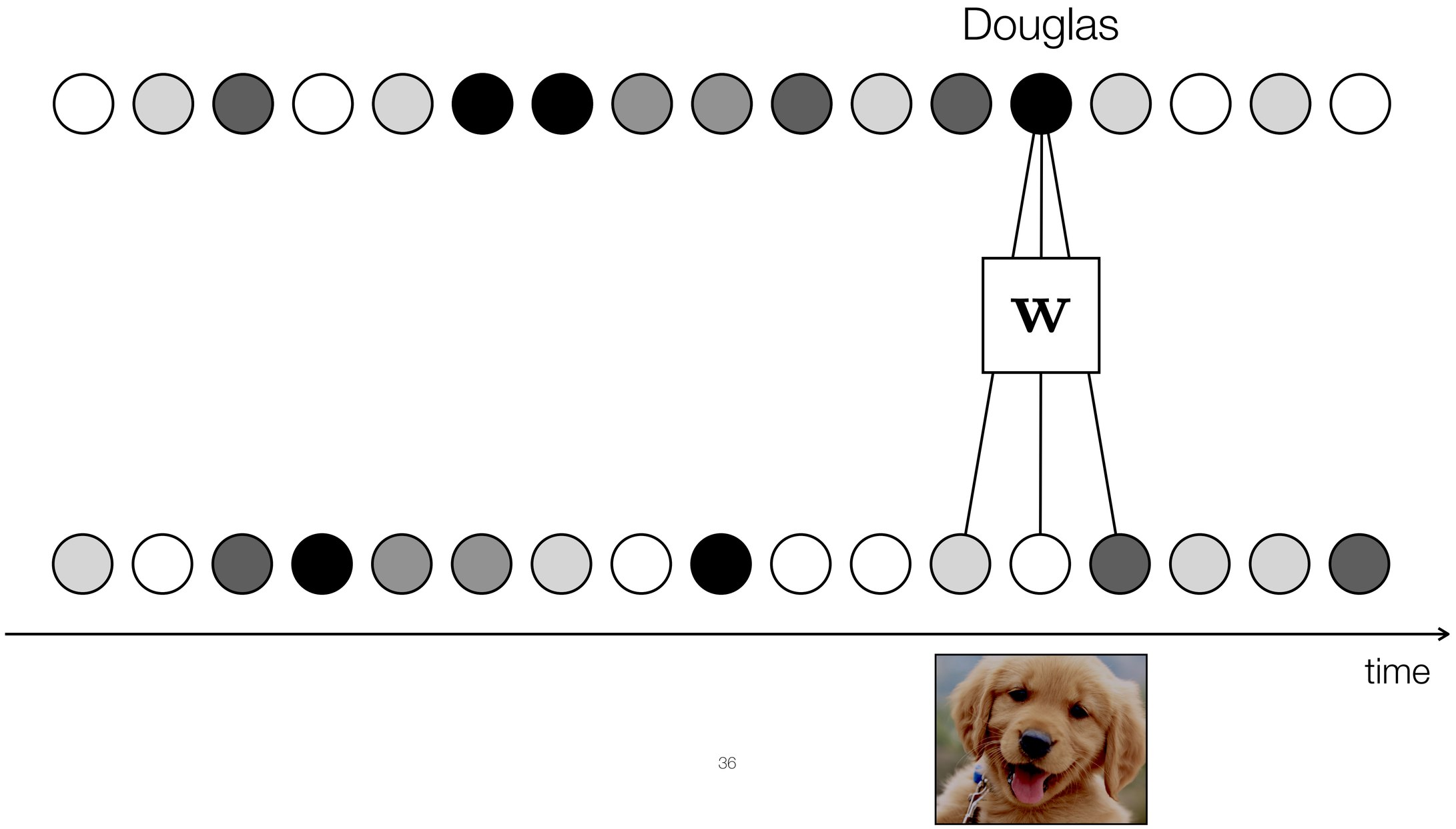
Lecture 12: Recurrent Models & Language



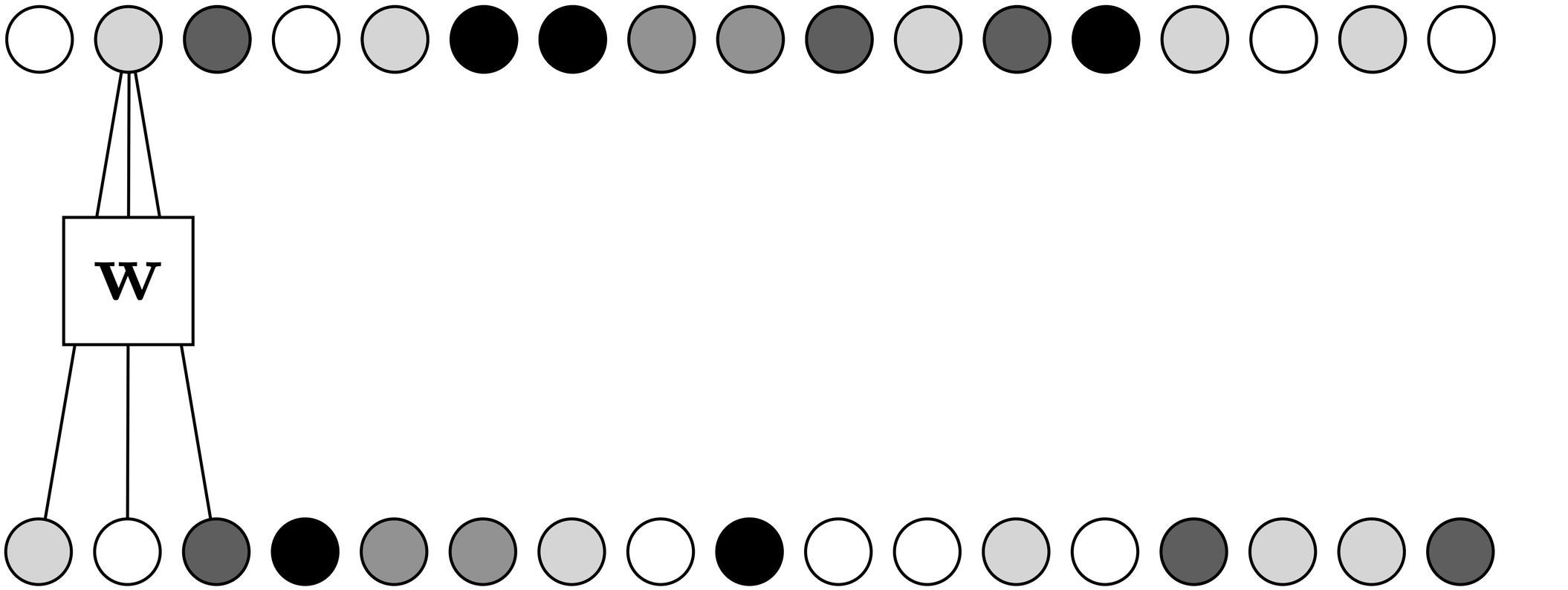
Source: Torralba, Freeman, Isola



Source: Torralba, Freeman, Isola

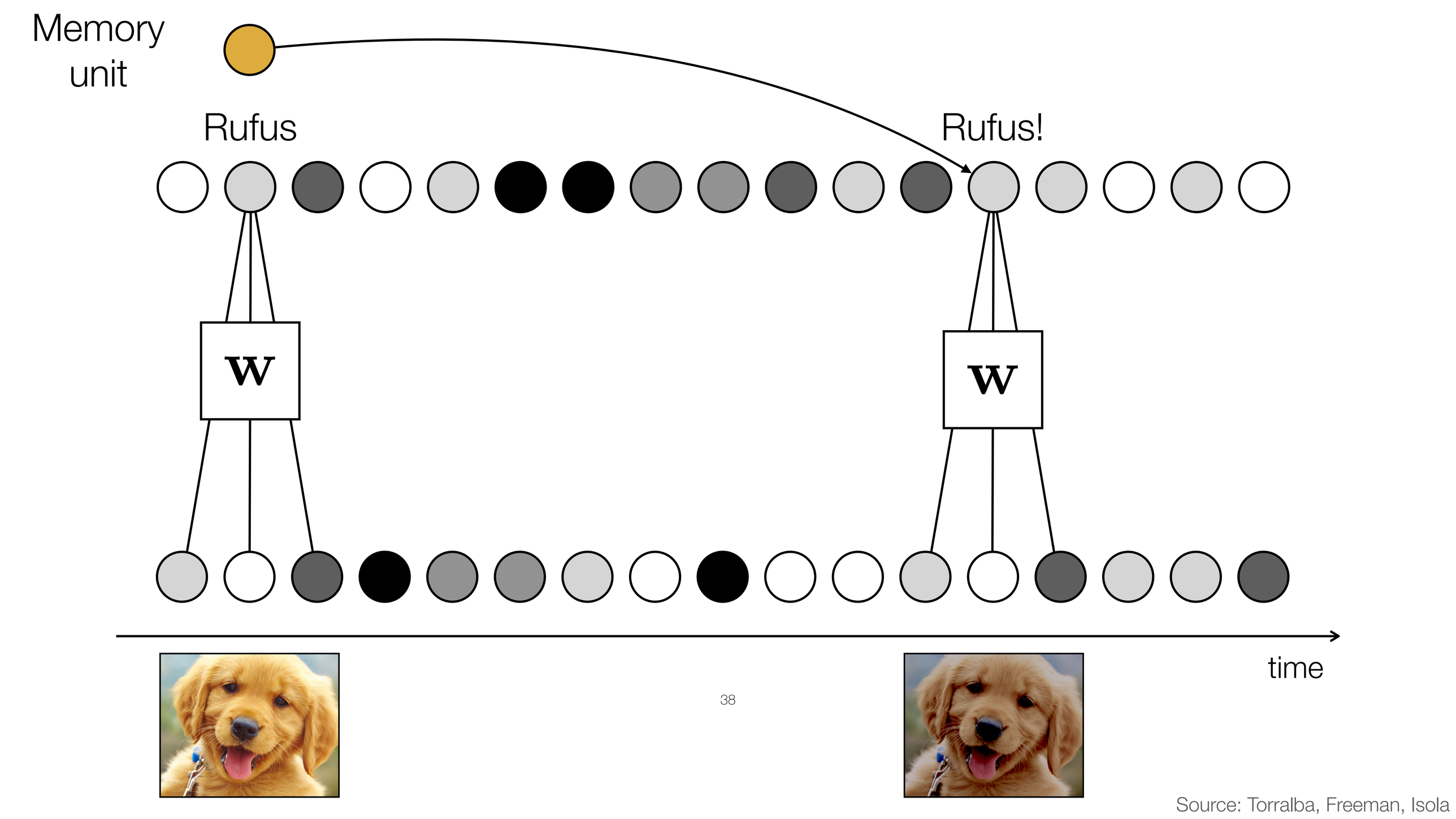
Memory unit

Rufus



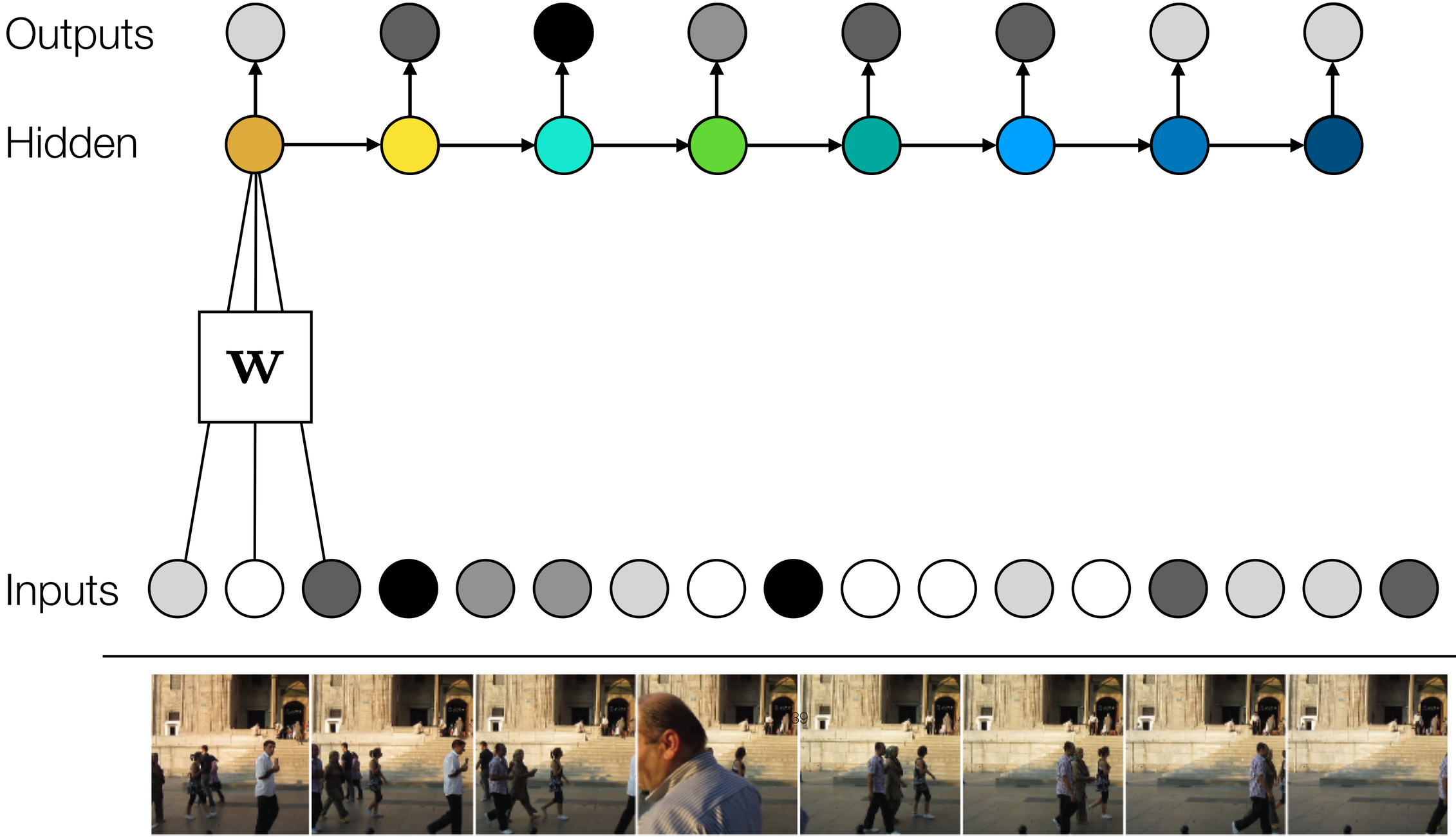
time

Source: Torralba, Freeman, Isola

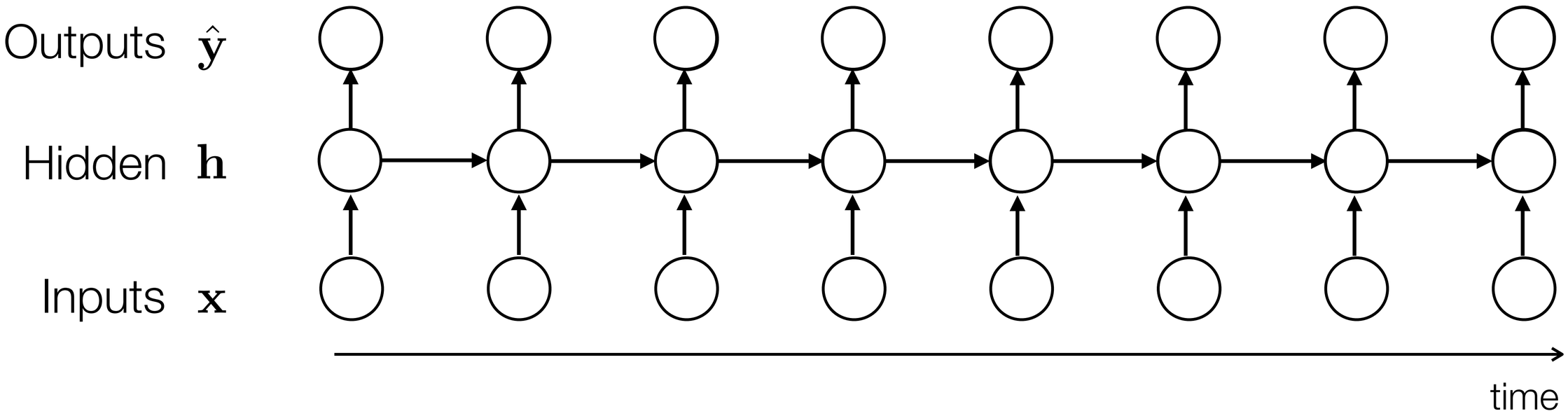


Source: Torralba, Freeman, Isola

Recurrent Neural Networks (RNNs)

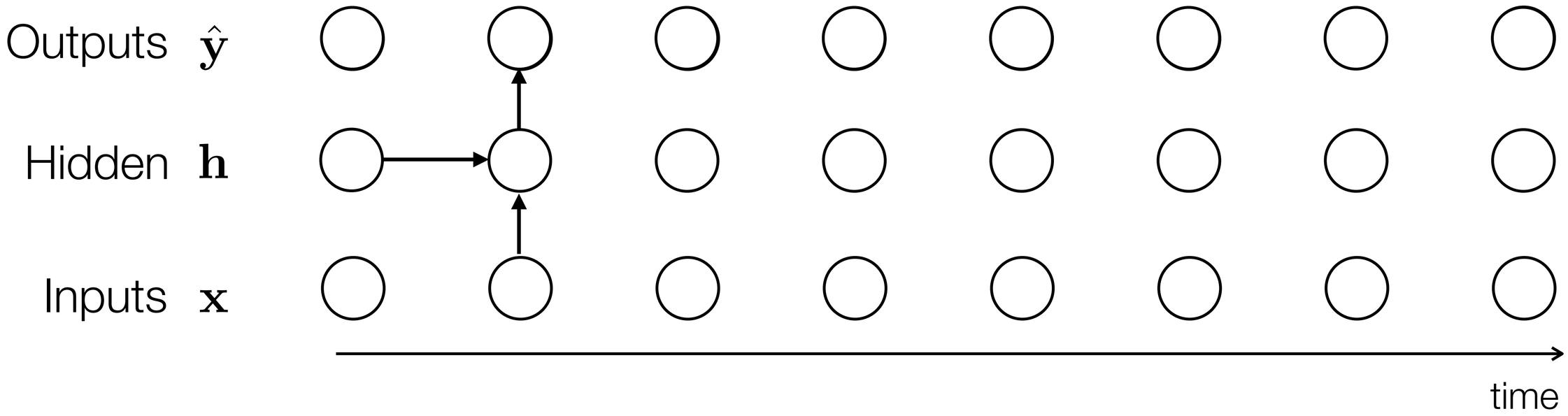


Recurrent Neural Networks (RNNs)



Source: Torralba, Freeman, Isola

Recurrent Neural Networks (RNNs)

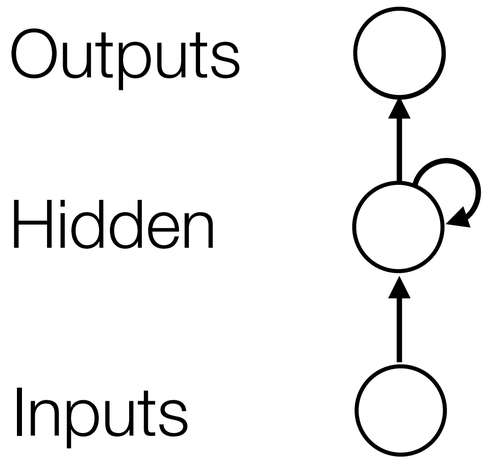


$$\mathbf{h}^{(t)} = f(\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)})$$

$$\mathbf{y}^{(t)} = g(\mathbf{h}^{(t)})$$

Source: Torralba, Freeman, Isola

Recurrent Neural Networks (RNNs)



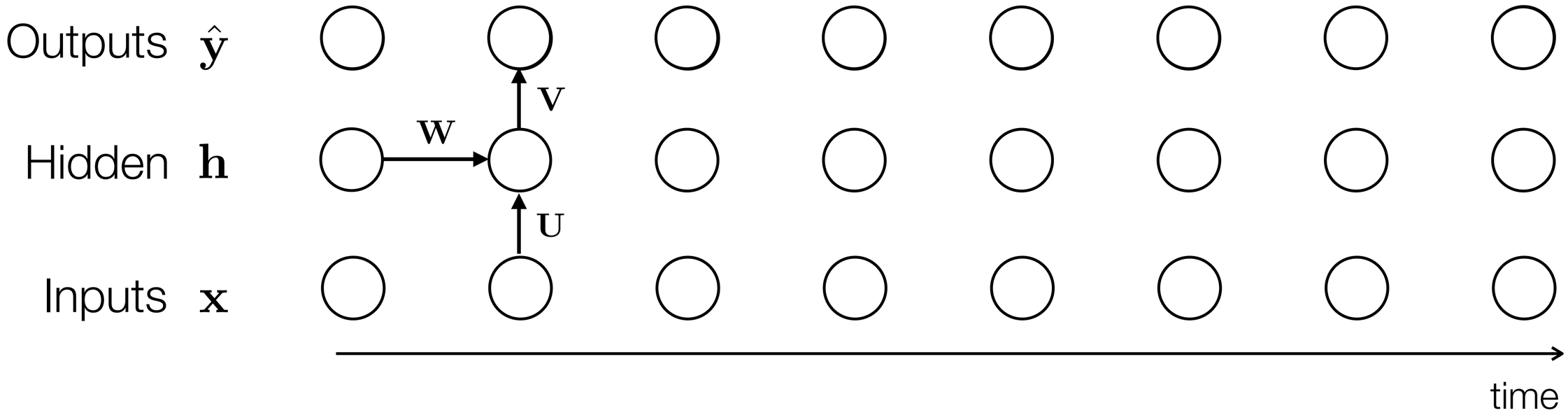
Recurrent!

$$\mathbf{h}^{(t)} = f(\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)})$$

$$\mathbf{y}^{(t)} = g(\mathbf{h}^{(t)})$$

Source: Torralba, Freeman, Isola

Recurrent Neural Networks (RNNs)



$$\mathbf{a}^{(t)} = \mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{U}\mathbf{x}^{(t)} + \mathbf{b}$$

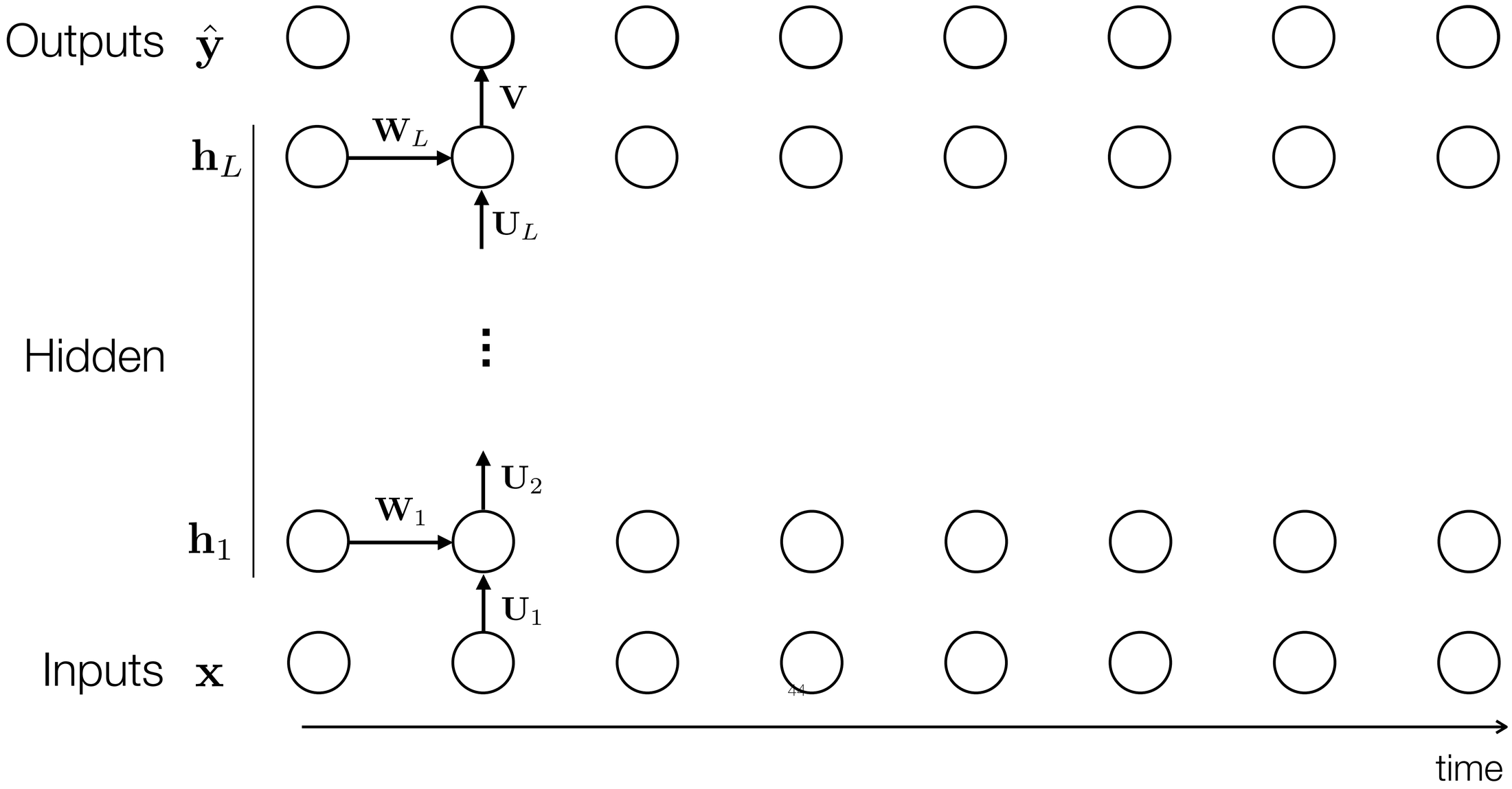
$$\mathbf{h}^{(t)} = \tanh(\mathbf{a}^{(t)})$$

$$\mathbf{o}^{(t)} = \mathbf{V}\mathbf{h}^{(t)} + \mathbf{c}$$

$$\hat{\mathbf{y}}^{(t)} = \texttt{softmax}(\mathbf{o}^{(t)})$$

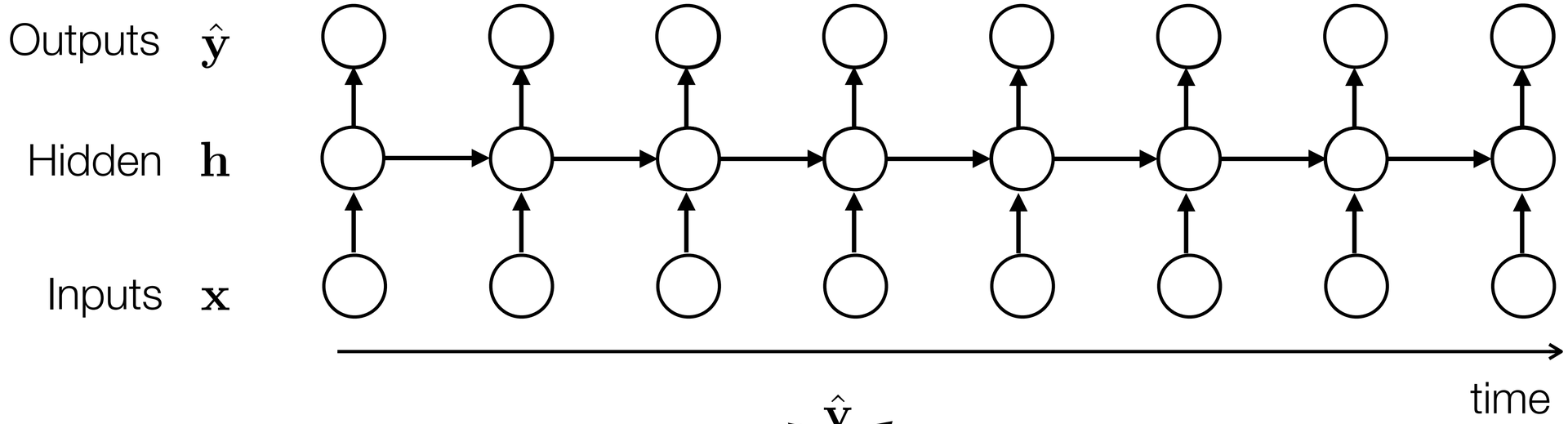
Source: Torralba, Freeman, Isola

Deep Recurrent Neural Networks (RNNs)

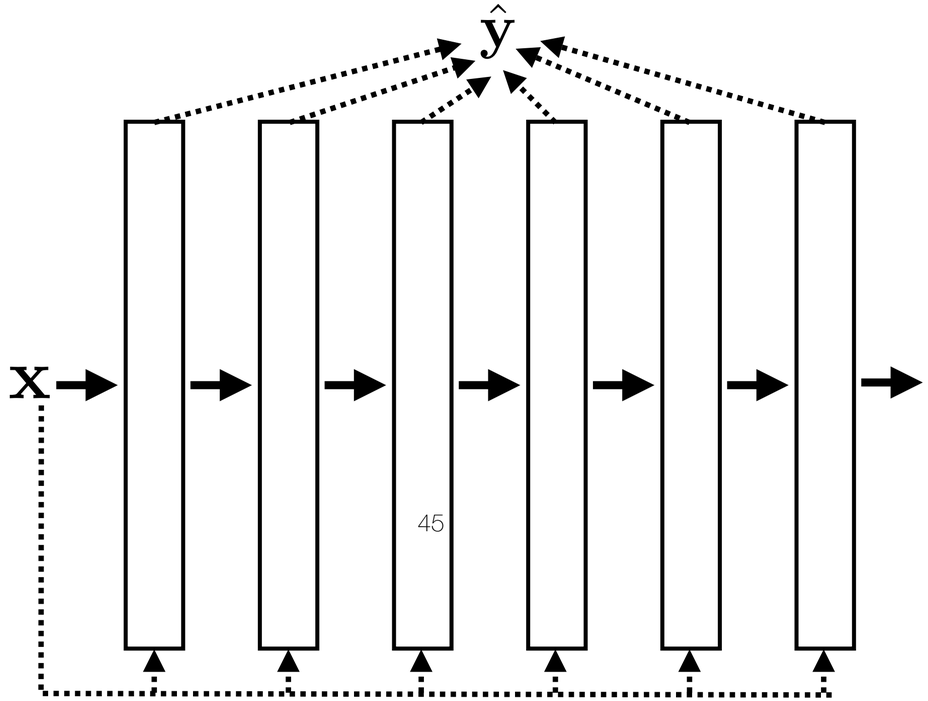


Source: Torralba, Freeman, Isola

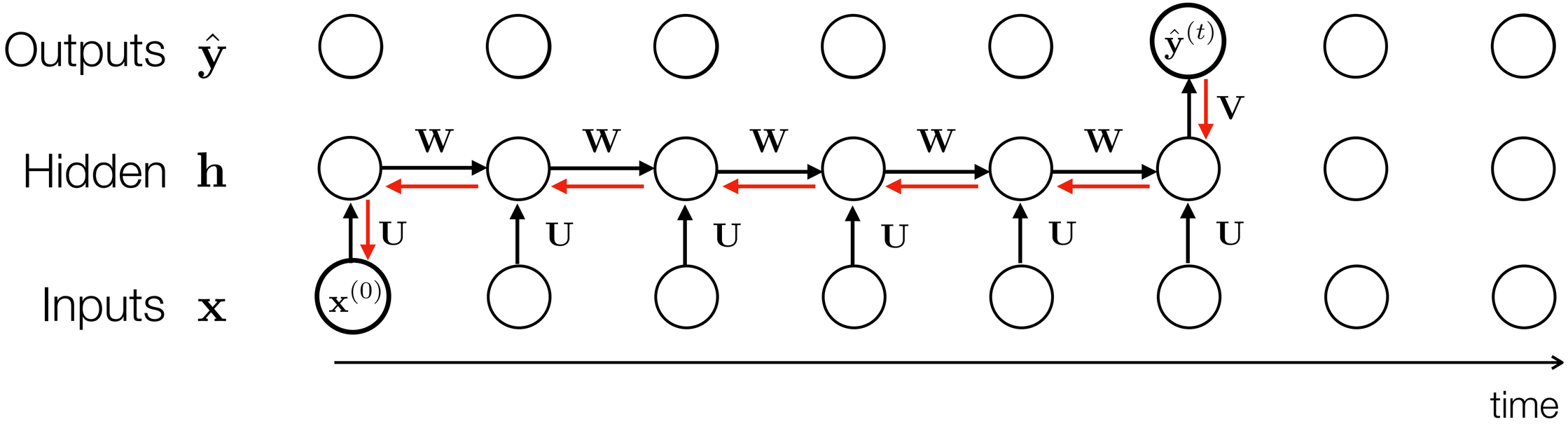
Unrolling an RNN



Equivalent to "unrolled" network with lots of layers



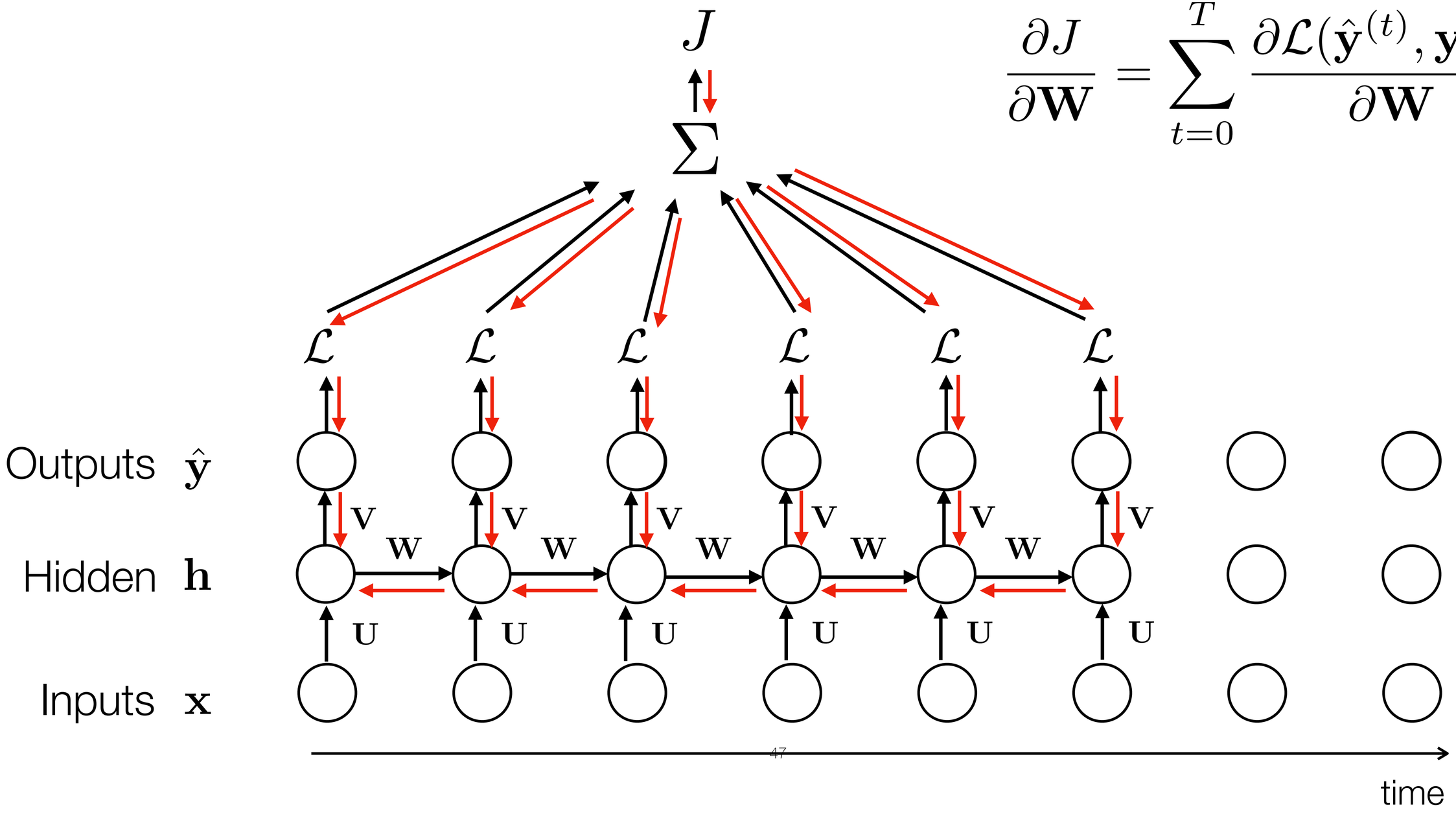
Backprop through time



$$\frac{\partial \hat{\mathbf{y}}^{(t)}}{\partial \mathbf{x}^{(0)}} = \frac{\partial \hat{\mathbf{y}}^{(t)}}{\partial \mathbf{h}^{(t)}} \frac{\partial \mathbf{h}^{(t)}}{\partial \mathbf{h}^{(t-1)}} \cdots \frac{\partial \mathbf{h}^{(1)}}{\partial \mathbf{h}^{(0)}} \frac{\partial \mathbf{h}^{(0)}}{\partial \mathbf{x}^{(0)}}$$

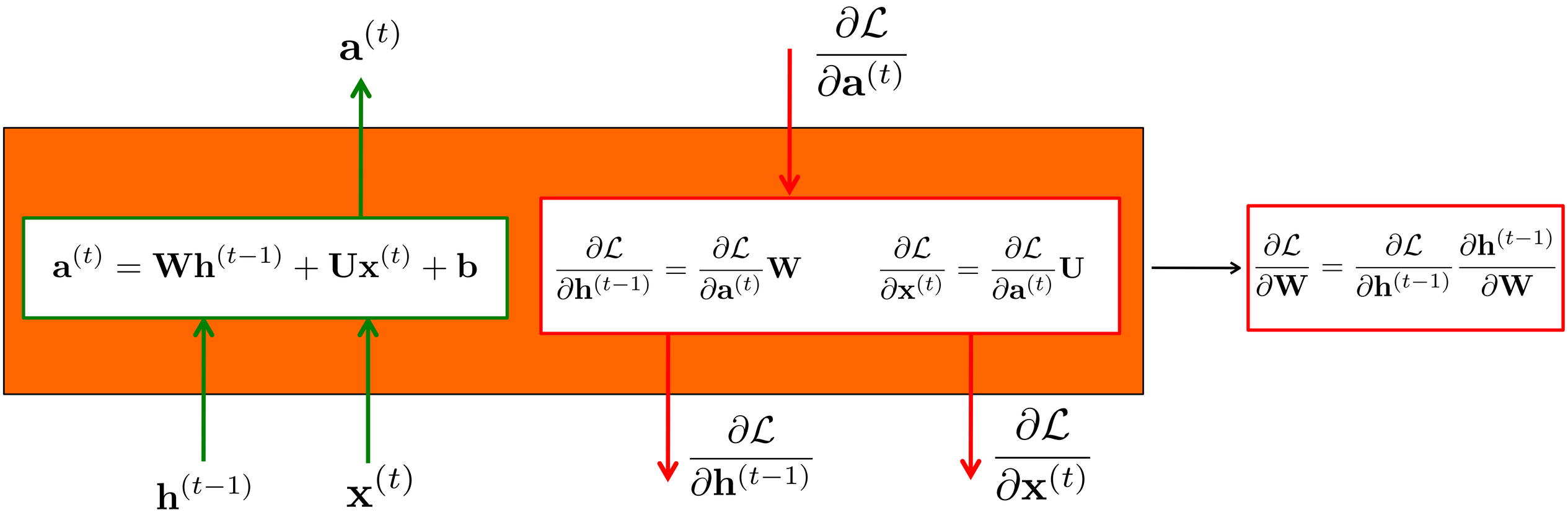
Source: Torralba, Freeman, Isola

$$\frac{\partial J}{\partial \mathbf{W}} = \sum_{t=0}^{T} \frac{\partial \mathcal{L}(\hat{\mathbf{y}}^{(t)}, \mathbf{y}^{(t)})}{\partial \mathbf{W}}$$



Source: Torralba, Freeman, Isola

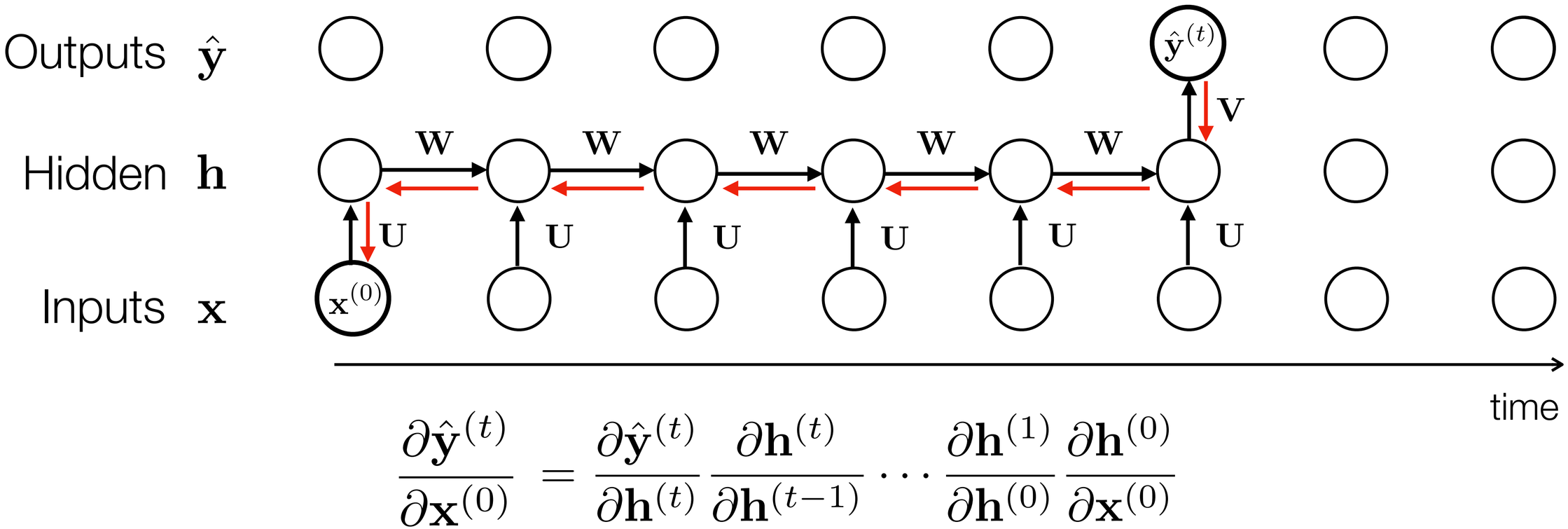
Recurrent linear layer



$$\frac{\partial J}{\partial \mathbf{W}} = \sum_{t=0}^{T} \frac{\partial \mathcal{L}(\hat{\mathbf{y}}^{(t)}, \mathbf{y}^{(t)})}{\partial \mathbf{W}}$$

Source: Torralba, Freeman, Isola

The problem of long-range dependences



$$\frac{\partial \hat{\mathbf{y}}^{(t)}}{\partial \mathbf{x}^{(0)}} = \frac{\partial \hat{\mathbf{y}}^{(t)}}{\partial \mathbf{h}^{(t)}} \frac{\partial \mathbf{h}^{(t)}}{\partial \mathbf{h}^{(t-1)}} \cdots \frac{\partial \mathbf{h}^{(1)}}{\partial \mathbf{h}^{(0)}} \frac{\partial \mathbf{h}^{(0)}}{\partial \mathbf{x}^{(0)}}$$

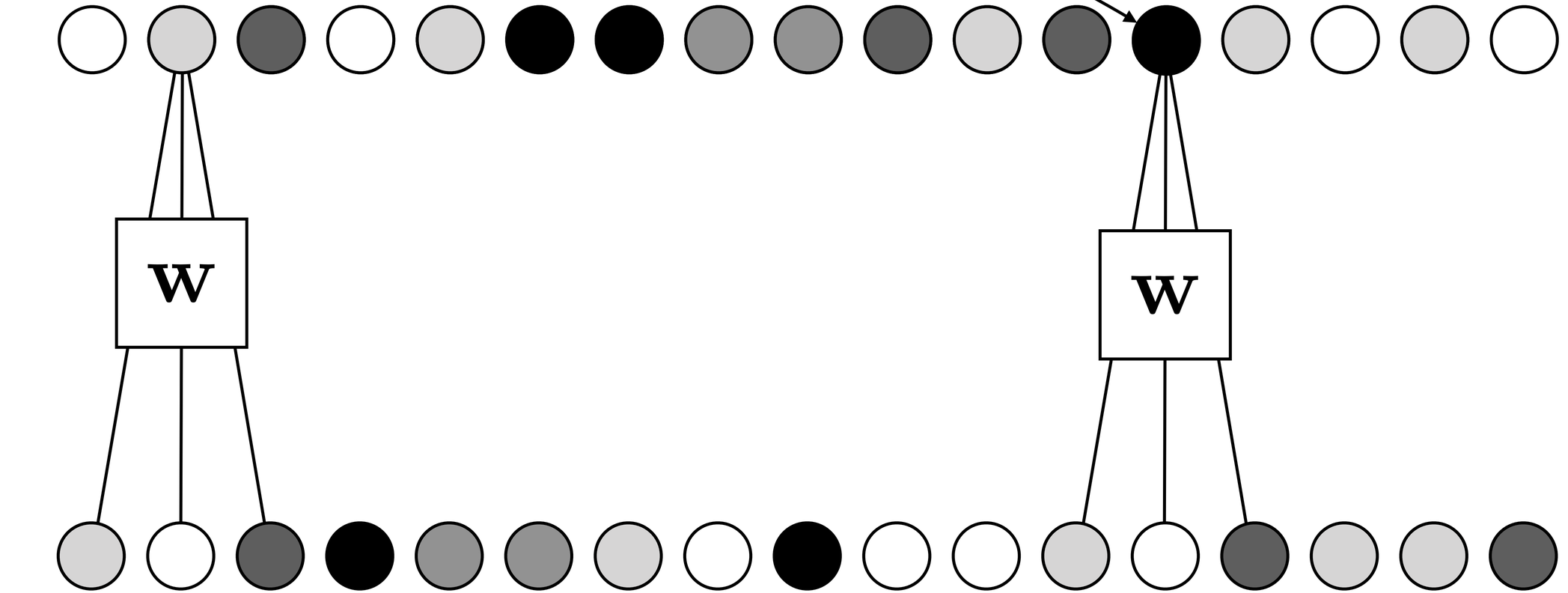
- Capturing long-range dependences requires propagating information through a long chain of dependences.
- Old observations are forgotten
- Stochastic gradients become high variance (noisy), and gradients may **vanish** or **explode**

Source: Torralba, Freeman, Isola

Memory unit

Rufus

Rufus!



time

Source: Torralba, Freeman, Isola

Memory units

...

W

W

time

Source: Torralba, Freeman, Isola

The problem of long-range dependences

Why not remember everything?

- Memory size grows with t
- This kind of memory is **nonparametric**: there is no finite set of parameters we can use to model it
- Markov-like assumption: future state only dependent on preceding hidden state. Very easy to forget things.
- By putting the right info in to the hidden state, RNNs can model dependencies that are arbitrarily far apart

Source: Torralba, Freeman, Isola

LSTMs

Long Short Term Memory

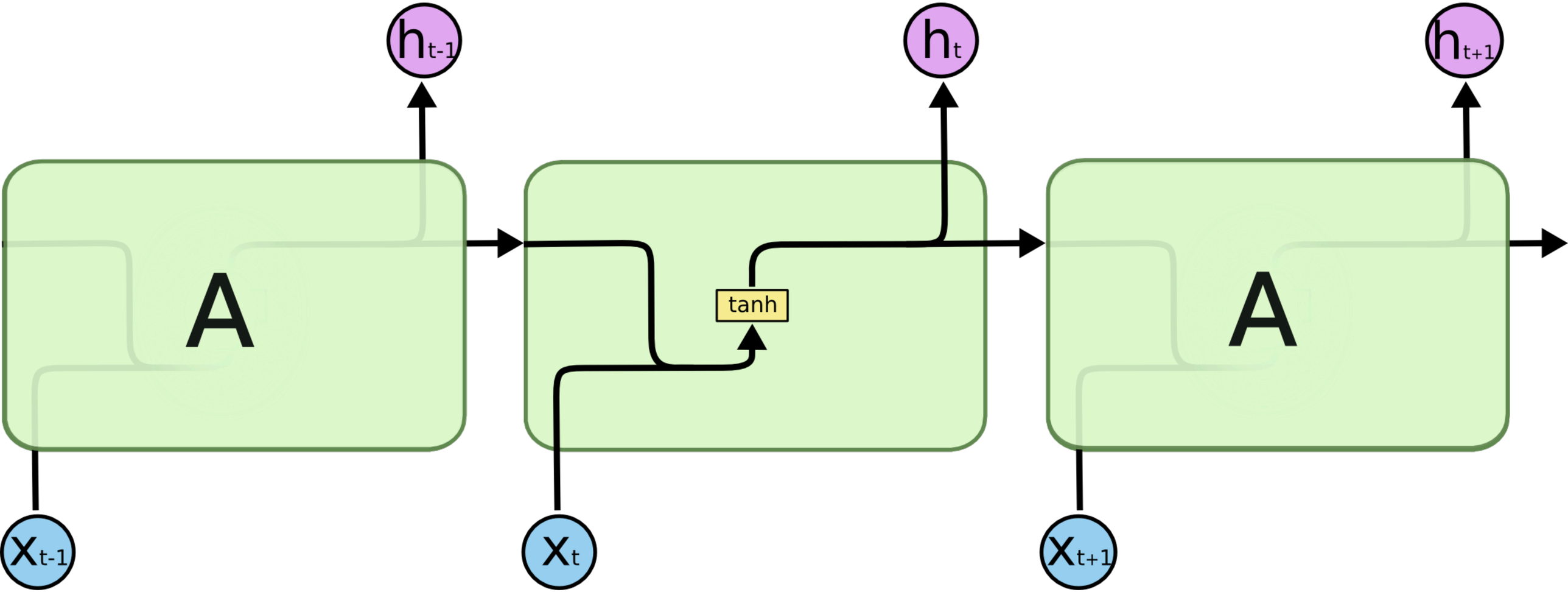
A special kind of RNN designed to avoid forgetting.

Related to ResNets inductive bias is that state transition is an identity function.

This way the default behavior is not to forget an old state. Instead of forgetting by default, the network has to *learn to forget*.

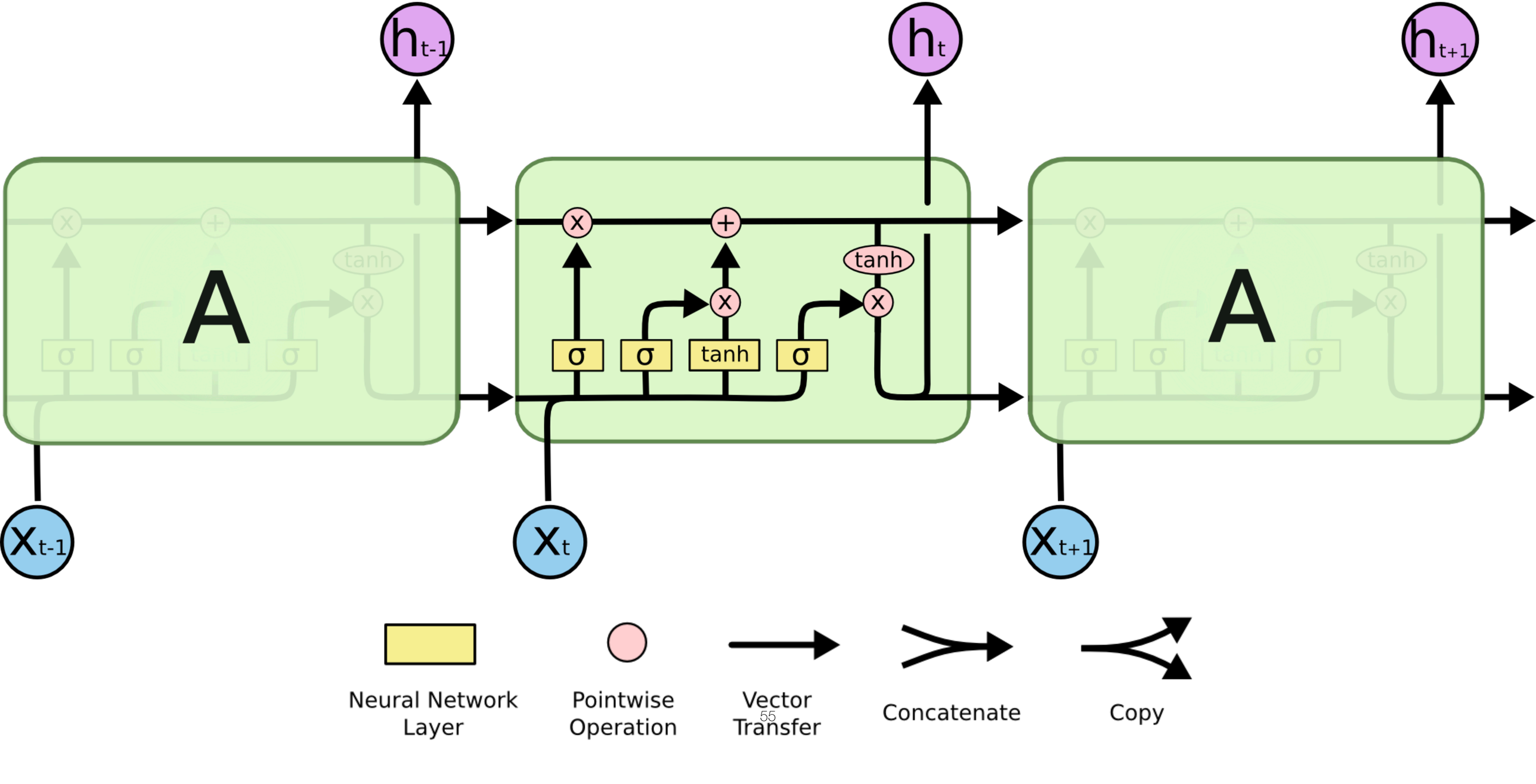
Bit of a complex design. Works well but simpler methods like Gated Recurrent Unit (GRU) are competitive [Jozefowicz et al. 2015].

Source: Torralba, Freeman, Isola



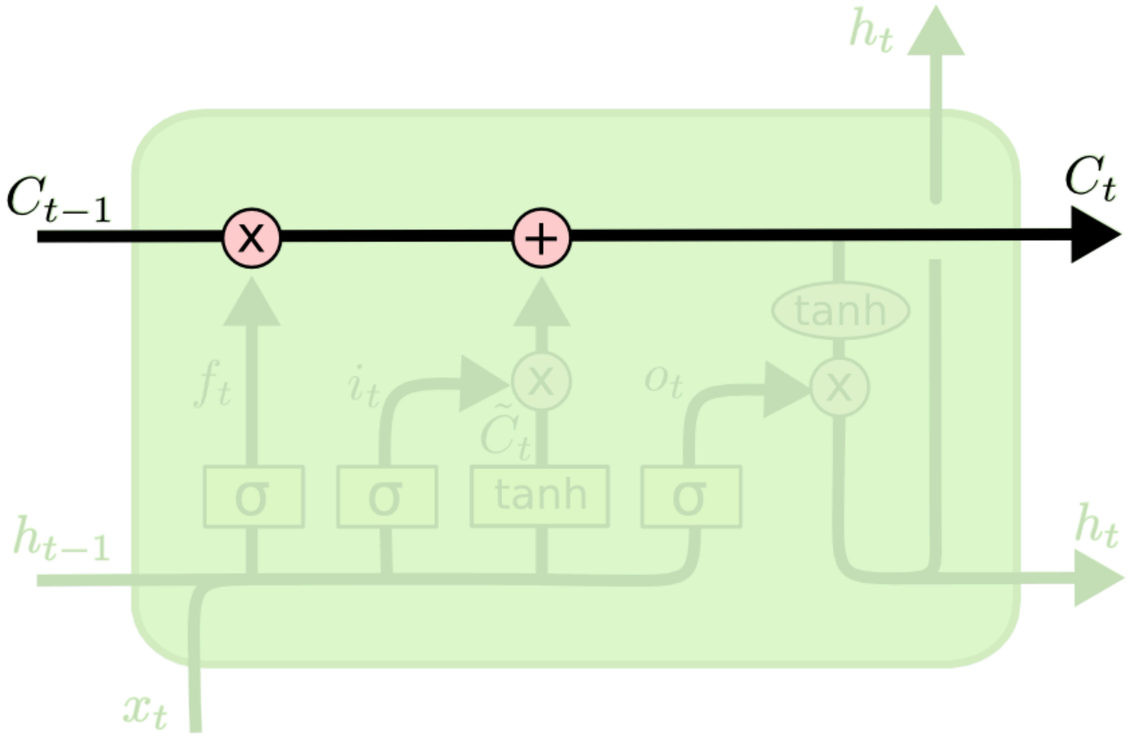
[Slide derived from Chris Olah: http://colah.github.io/posts/2015-08-Understanding-LSTMs/]

Source: Torralba, Freeman, Isola



[Slide derived from Chris Olah: http://colah.github.io/posts/2015-08-Understanding-LSTMs/]

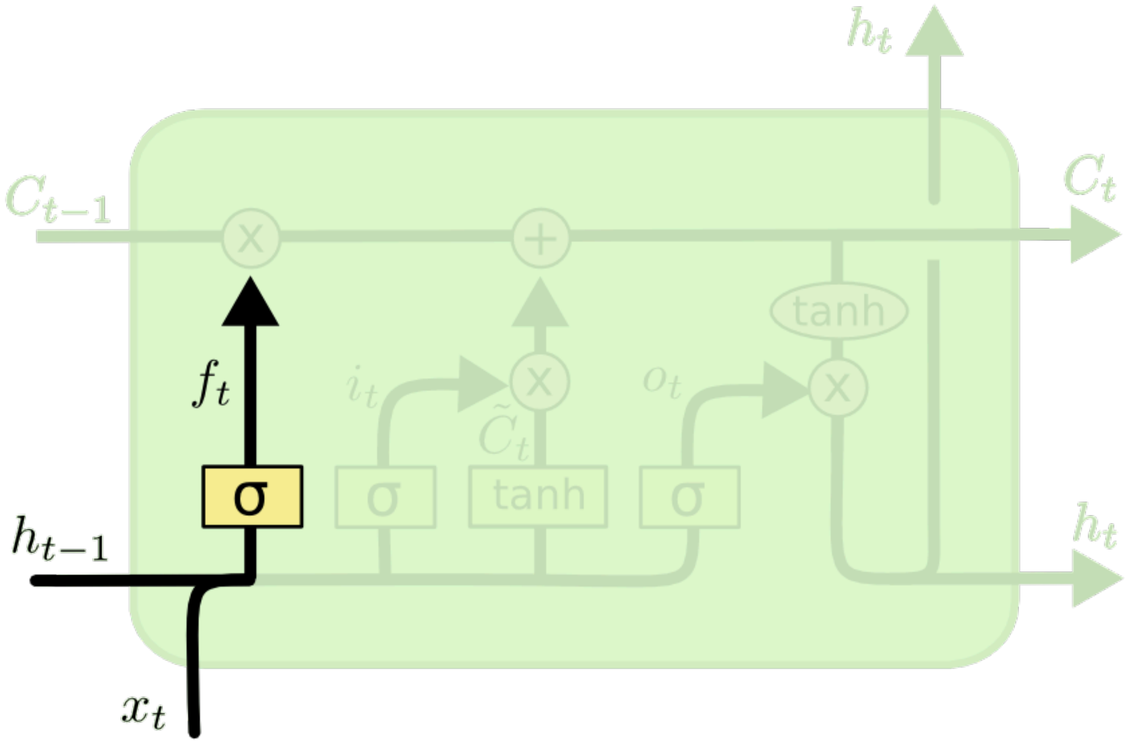
Source: Torralba, Freeman, Isola

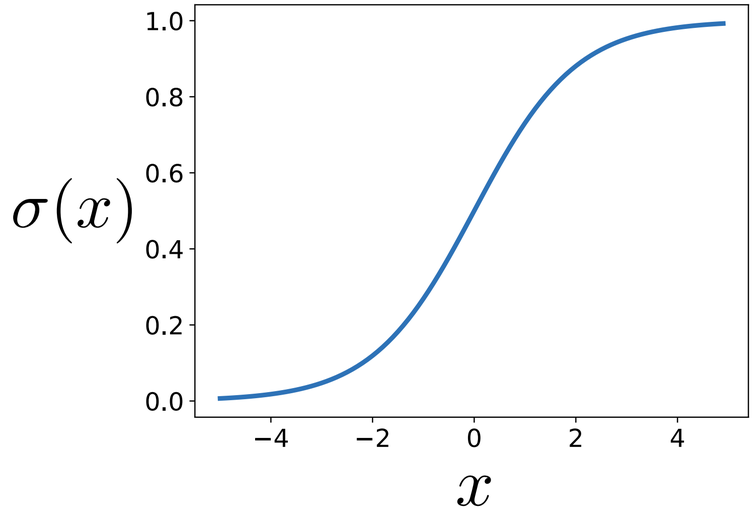


C_t = Cell state

[Slide derived from Chris Olah: http://colah.github.io/posts/2015-08-Understanding-LSTMs/]

Source: Torralba, Freeman, Isola





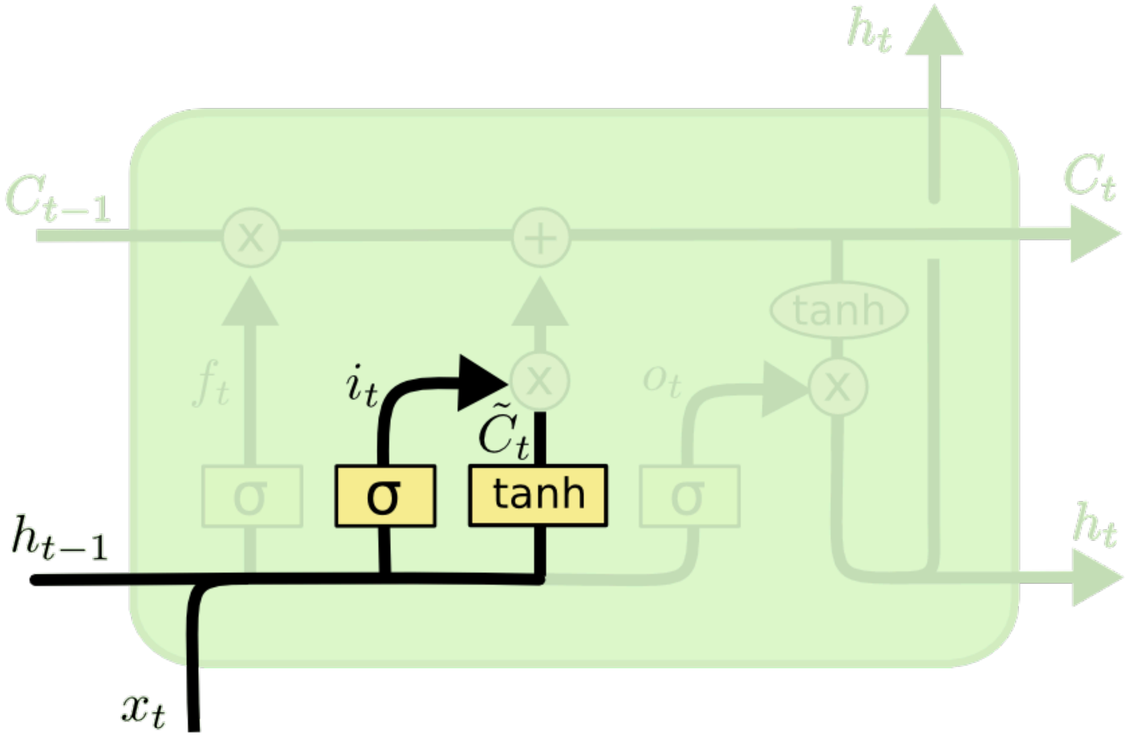
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Decide what information to throw away from the cell state.

Each element of cell state is multiplied by ~1 (remember) or ~0 (forget).

[Slide derived from Chris Olah: http://colah.github.io/posts/2015-08-Understanding-LSTMs/]

Source: Torralba, Freeman, Isola



which indices to write to

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

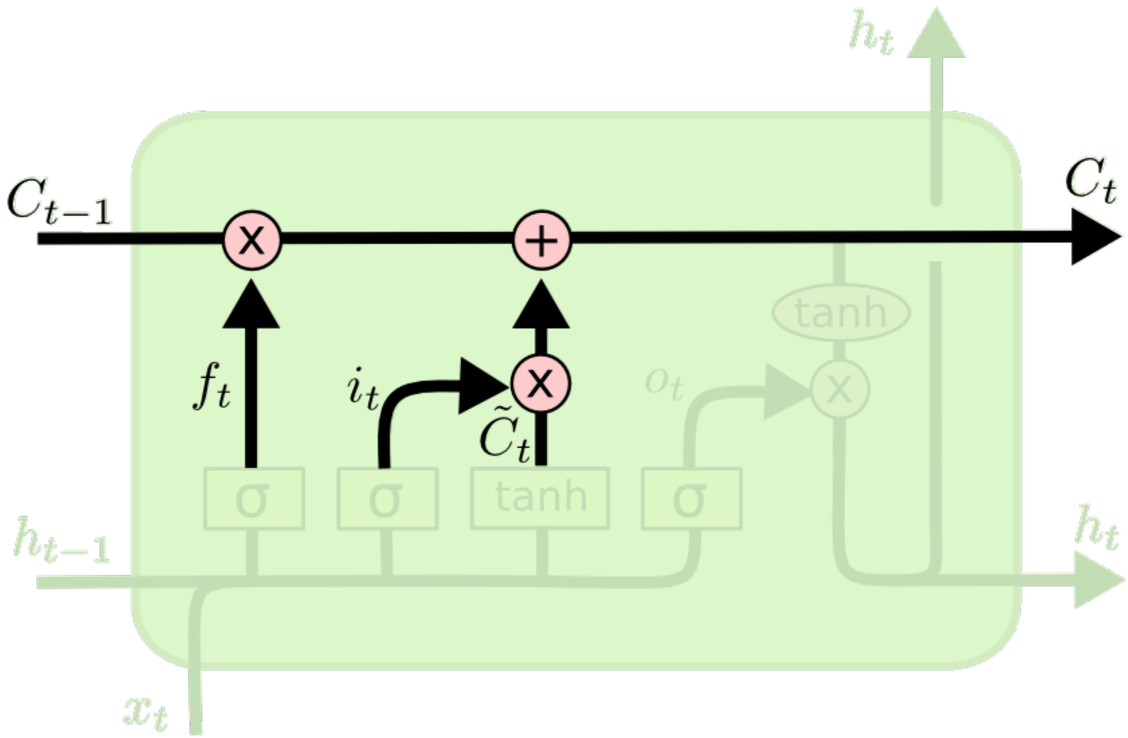
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

what to write to those indices

Decide what new information to add to the cell state.

[Slide derived from Chris Olah: http://colah.github.io/posts/2015-08-Understanding-LSTMs/]

Source: Torralba, Freeman, Isola

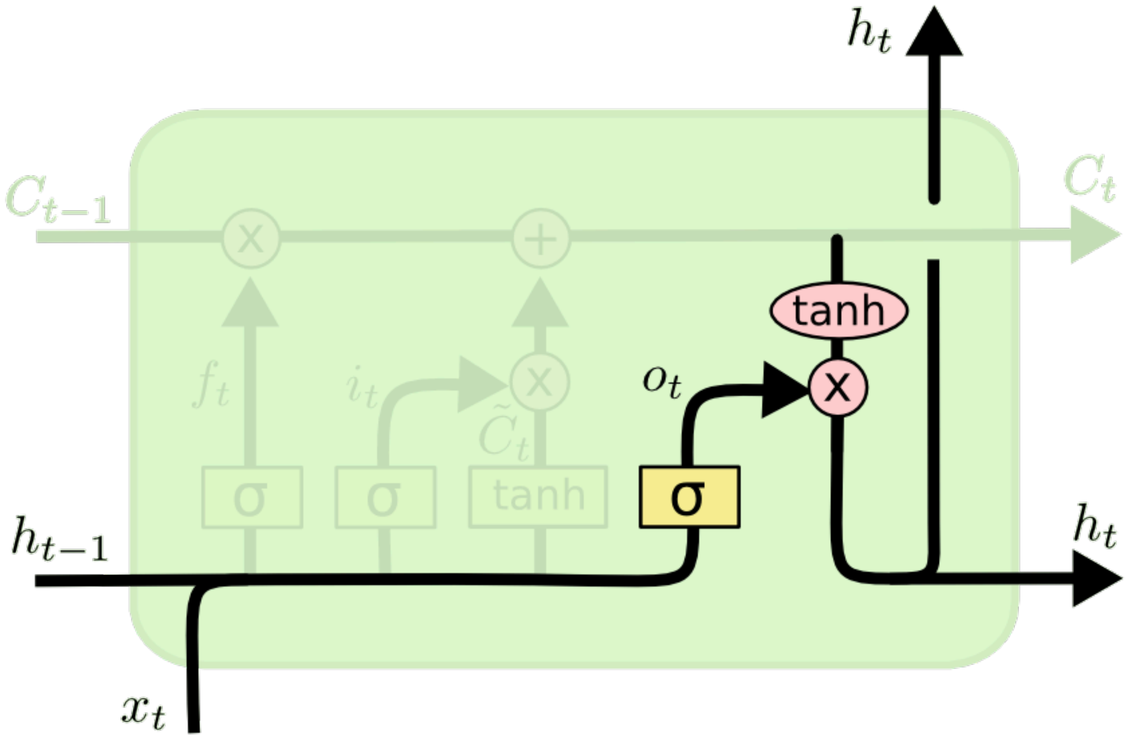


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Forget selected old information, write selected new information.

[Slide derived from Chris Olah: http://colah.github.io/posts/2015-08-Understanding-LSTMs/]

Source: Torralba, Freeman, Isola



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

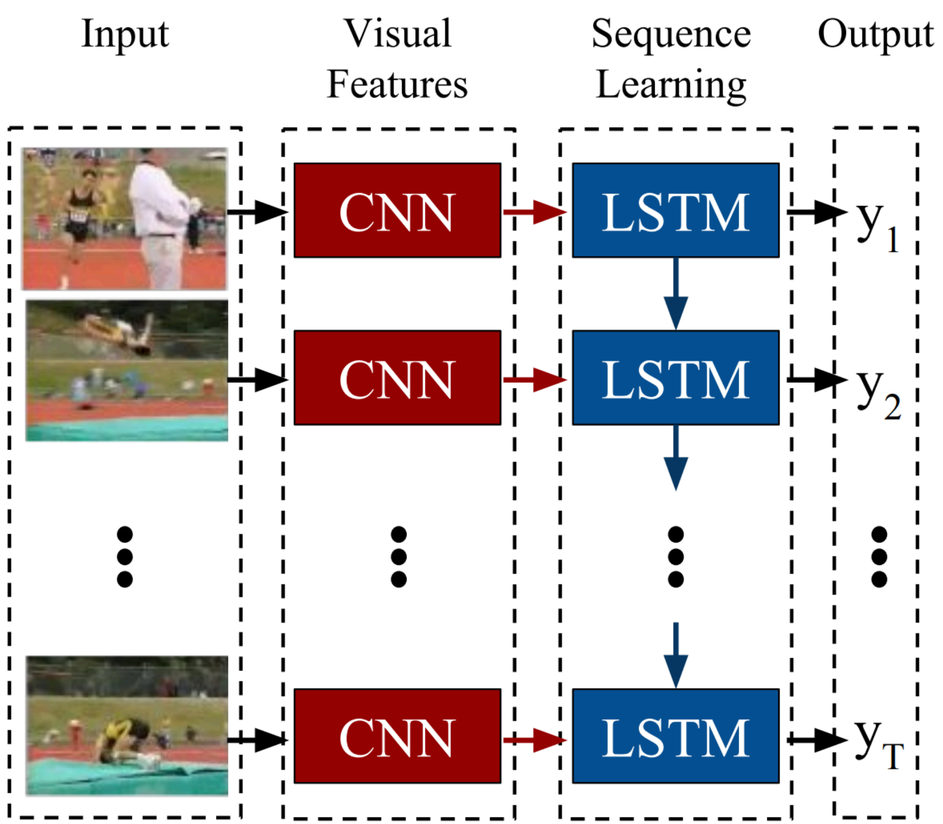
$$h_t = o_t * \tanh\left(C_t\right)$$

After having updated the cell state's information, decide what to output.

[Slide derived from Chris Olah: http://colah.github.io/posts/2015-08-Understanding-LSTMs/]

Source: Torralba, Freeman, Isola

Some uses for LSTMs



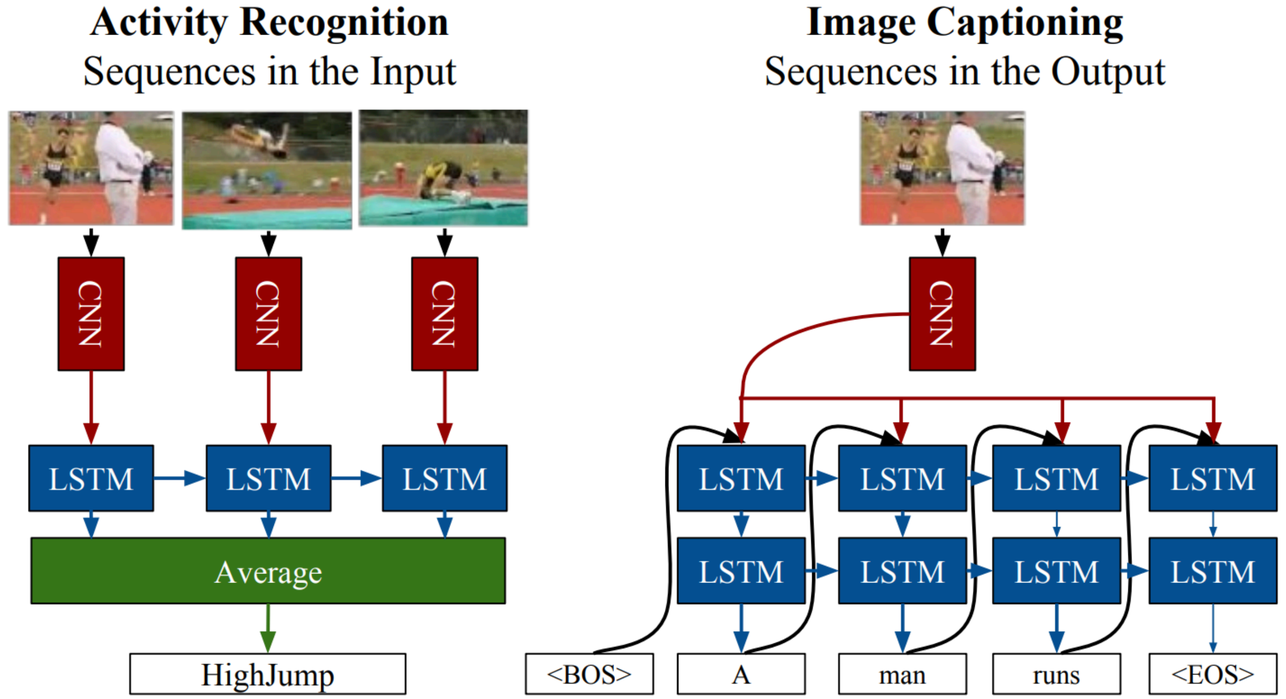


Figure from [Donahue et al. 2016]

The problem of long-range dependences

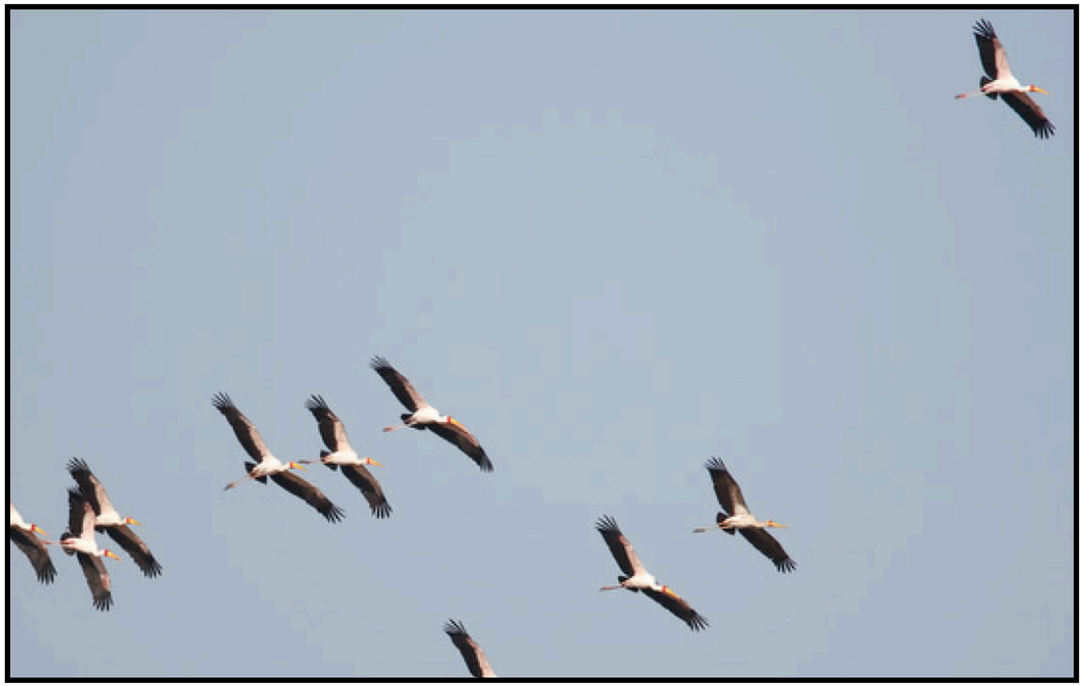
Other methods exist that do directly link old "memories" (observations or hidden states) to future predictions:

- Temporal convolutions (but temporal horizon often short)
- Attention (see https://arxiv.org/abs/1706.03762)
- Memory networks (see https://arxiv.org/abs/1410.3916)

Source: Torralba, Freeman, Isola

Language

Image captioning

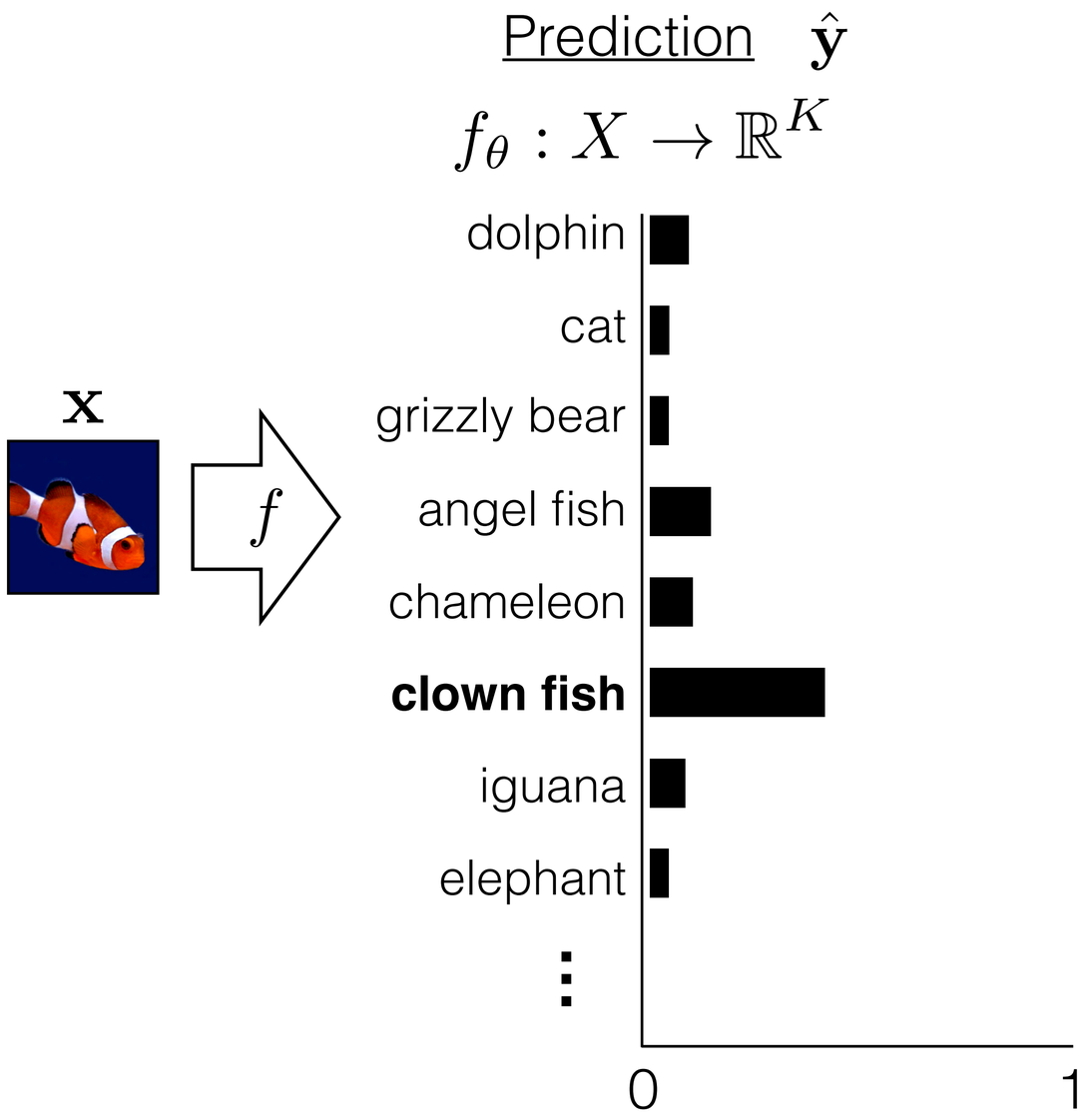


f

“A flock of birds against a gray sky”

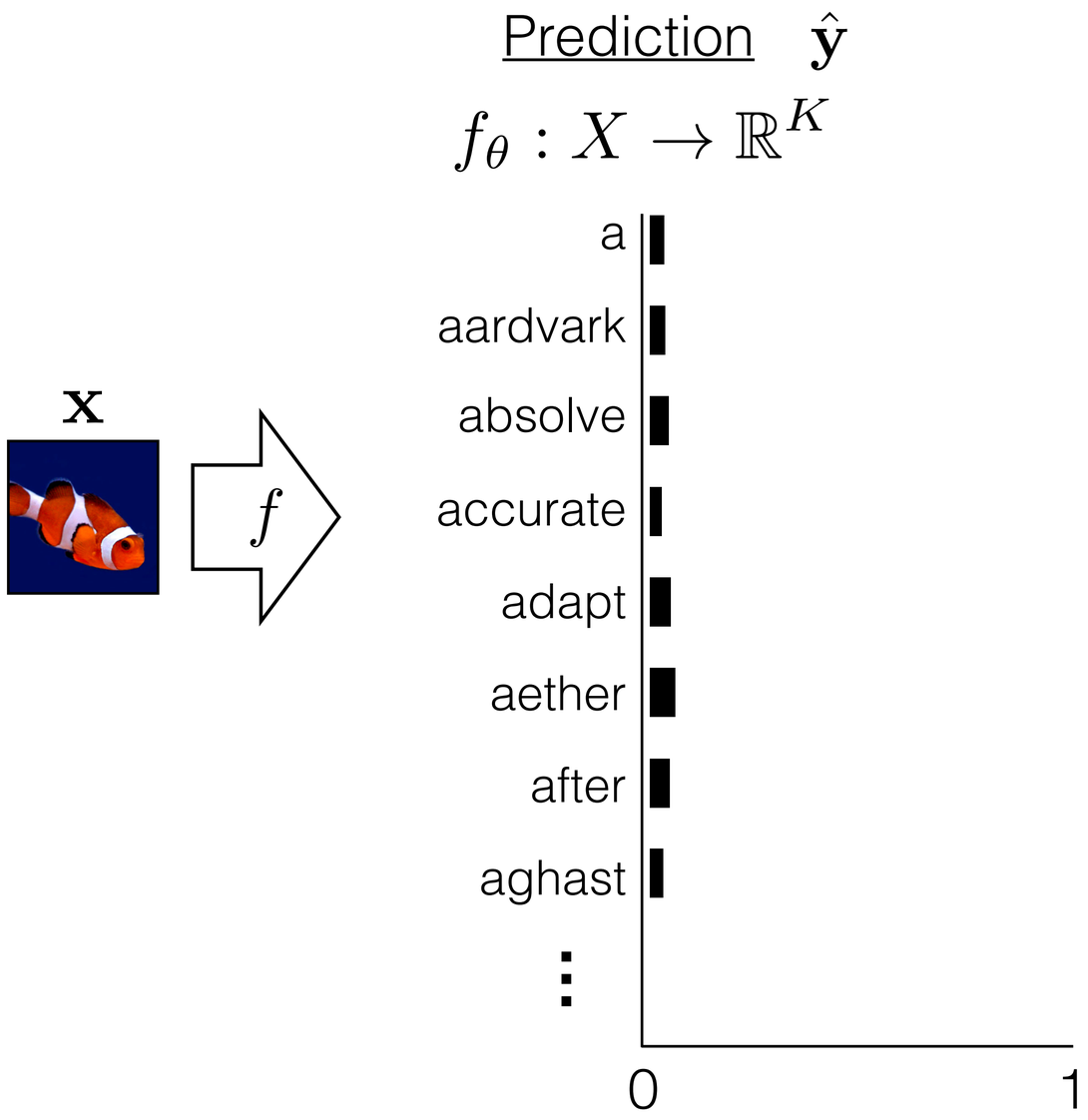
Source: Isola, Torralba, Freeman

How to represent words as numbers?



Source: Isola, Torralba, Freeman

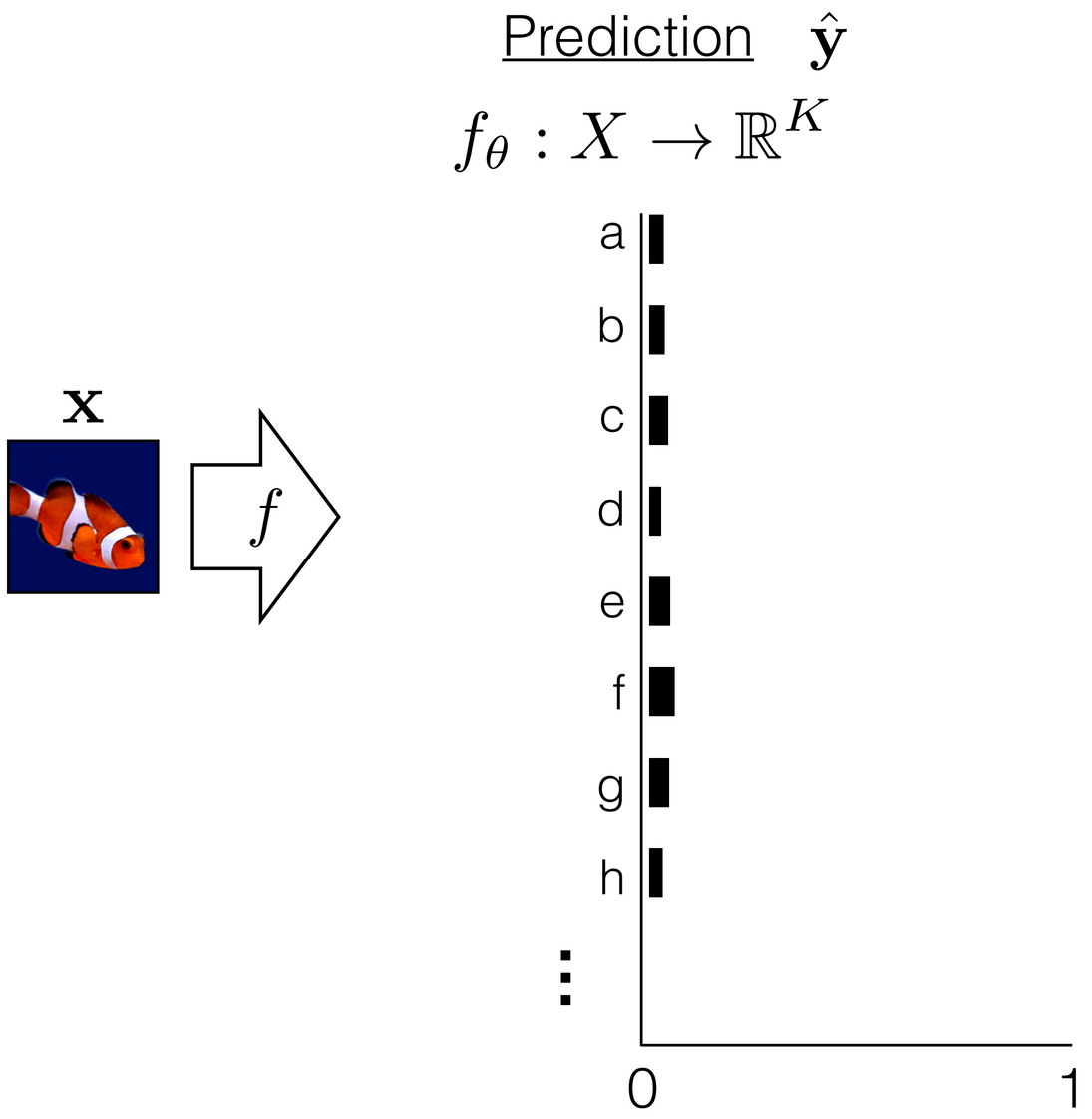
How to represent words as numbers?



Rather than having just a handful of possible object classes, we can represent all words in a large vocabulary using a very large K (e.g., K=100,000).

Source: Isola, Torralba, Freeman

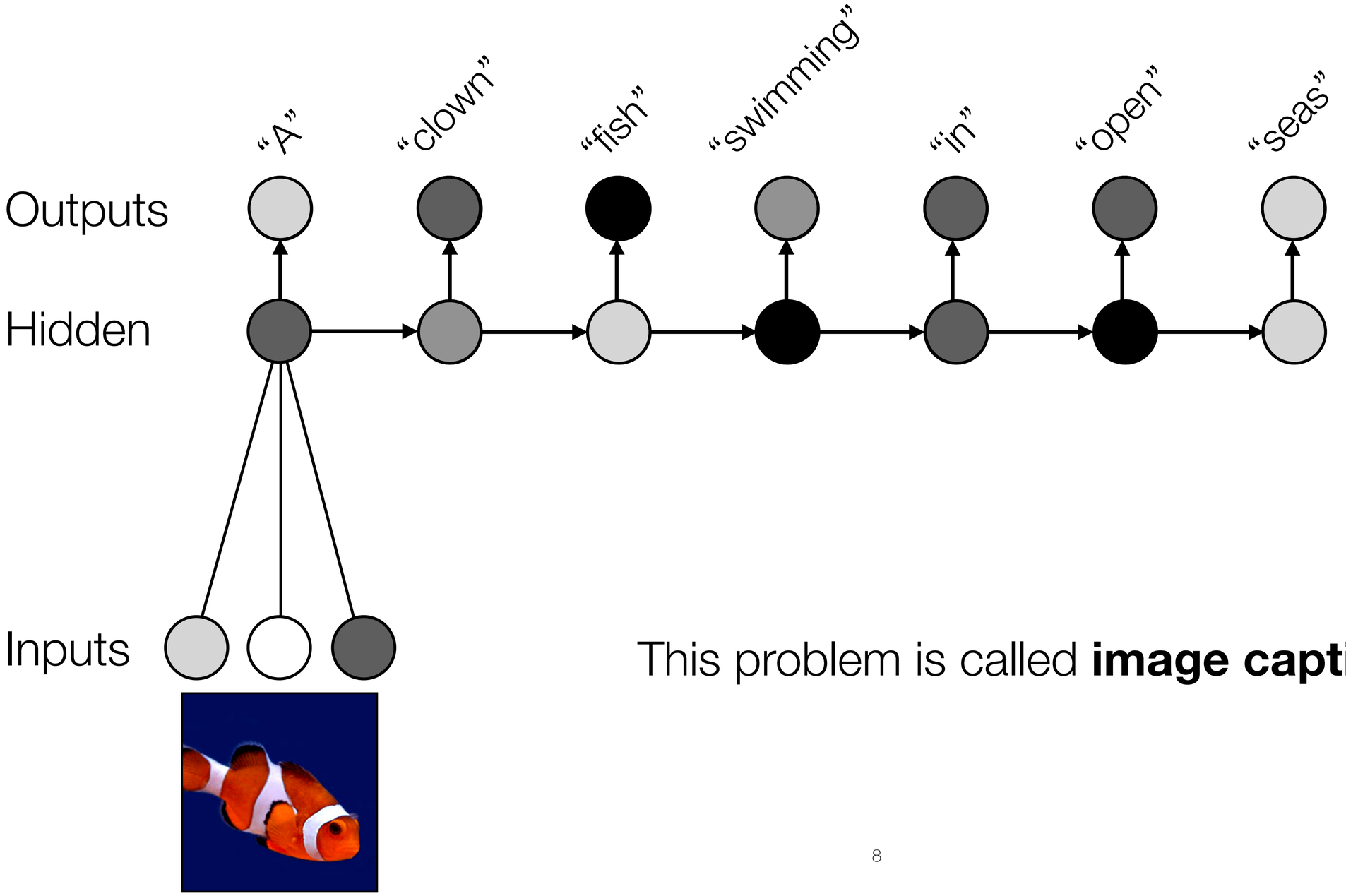
How to represent words as numbers?



Or, represent each character as a class (e.g., K=26 for English letters),

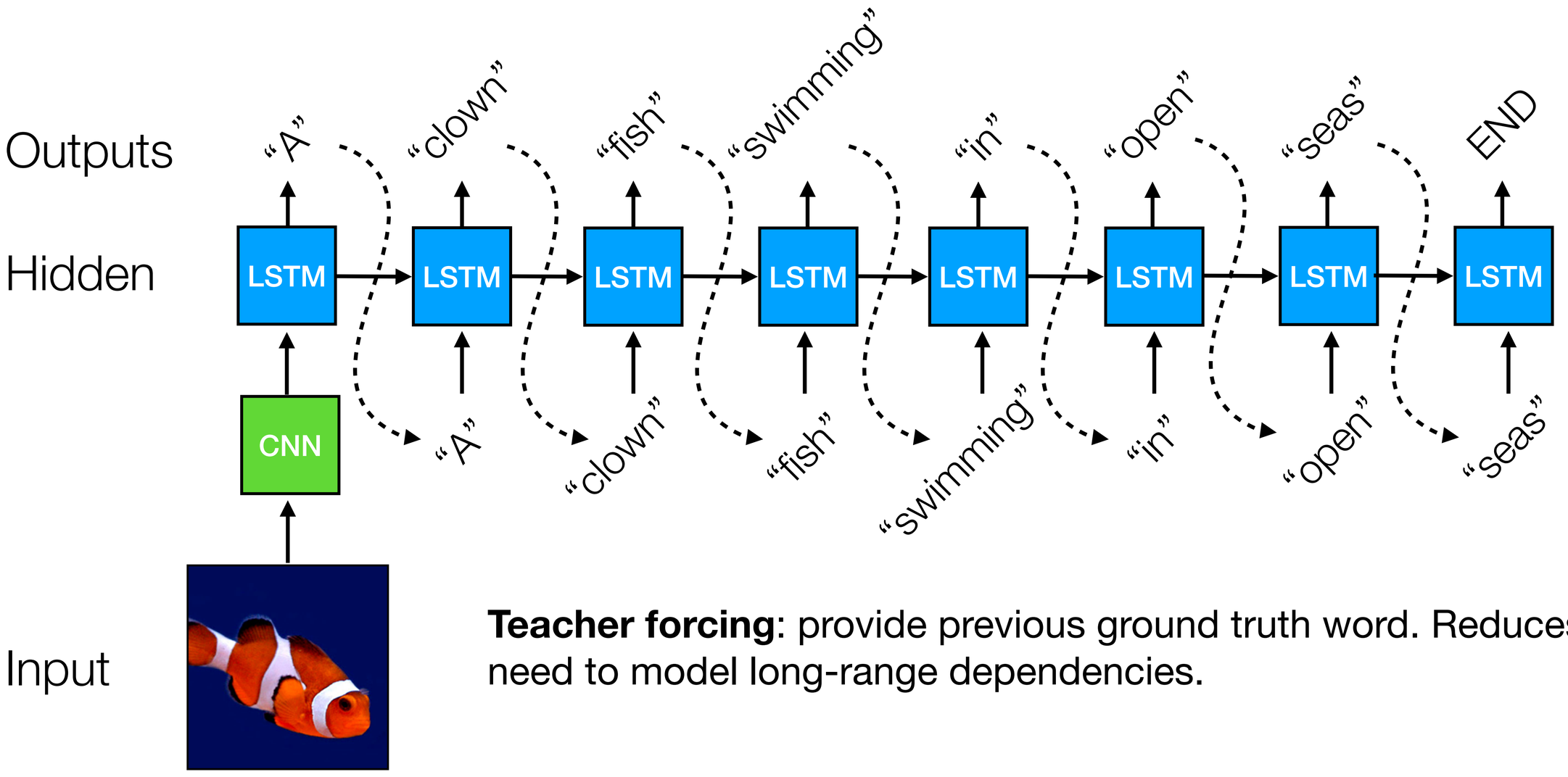
and represent words as a sequence of characters.

Source: Isola, Torralba, Freeman



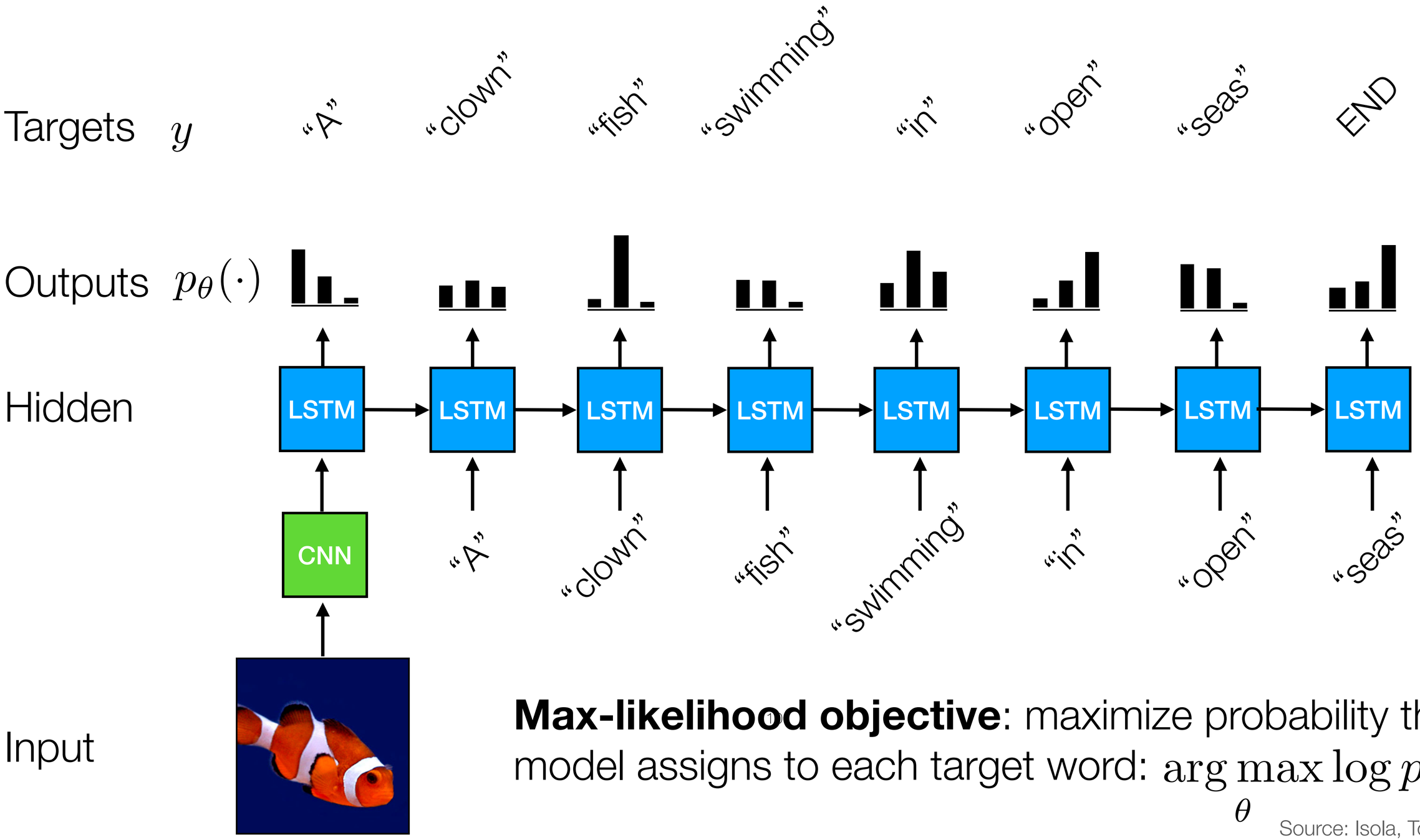
This problem is called **image captioning.**

Source: Isola, Torralba, Freeman



Teacher forcing: provide previous ground truth word. Reduces need to model long-range dependencies.

Source: Isola, Torralba, Freeman



Max-likelihood objective: maximize probability the model assigns to each target word: $\arg\max_\theta \log p_\theta(y)$

Source: Isola, Torralba, Freeman

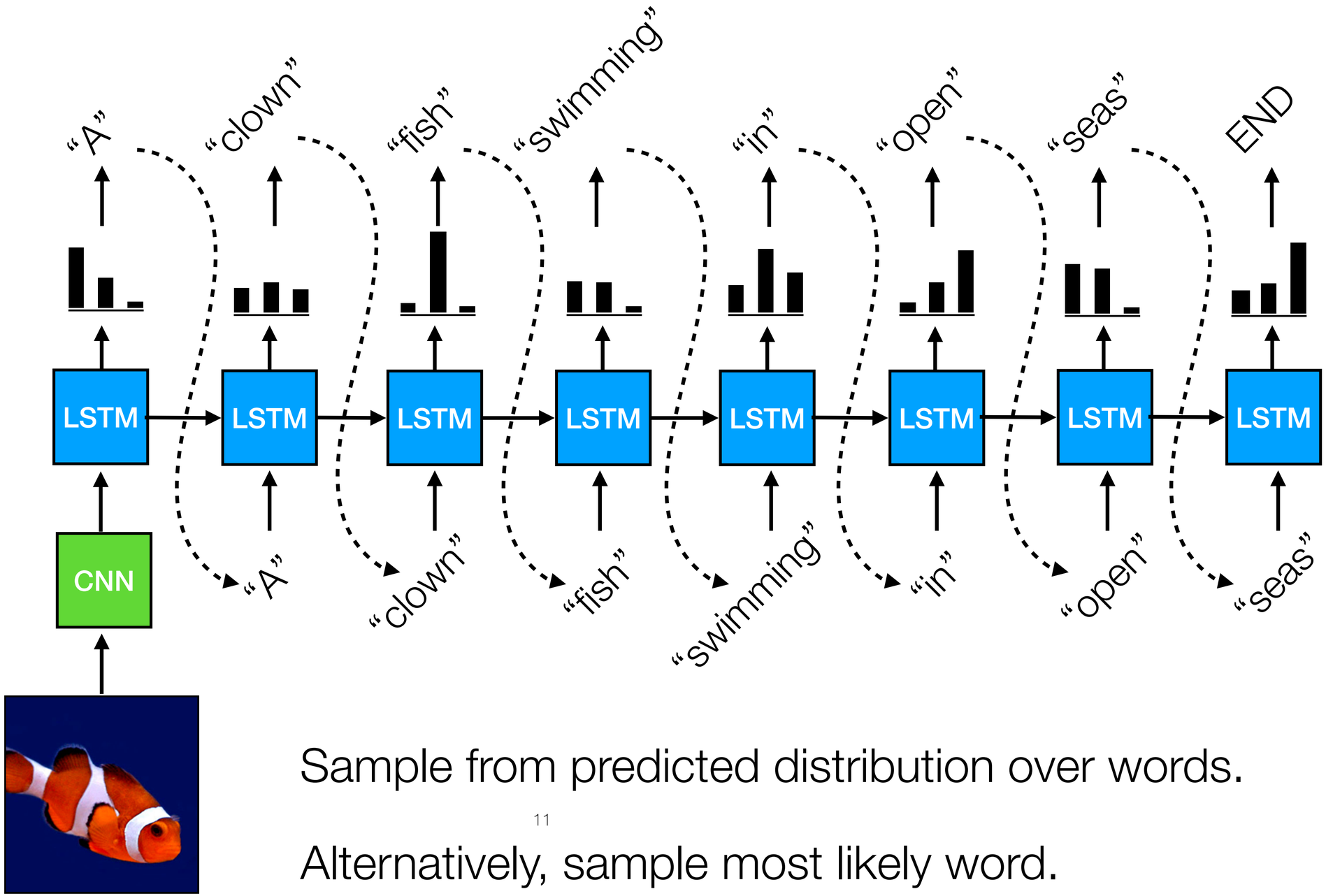
Testing

Samples

Outputs $p_\theta(\cdot)$

Hidden

Input

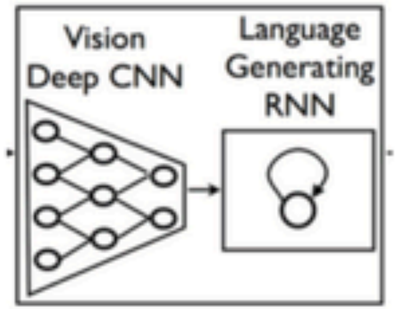


Sample from predicted distribution over words.

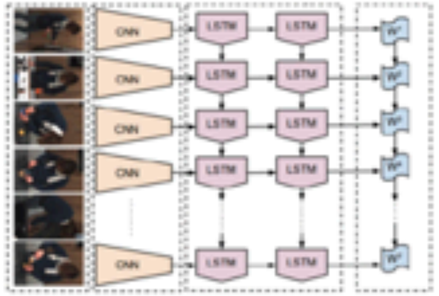
Alternatively, sample most likely word.

Source: Isola, Torralba, Freeman

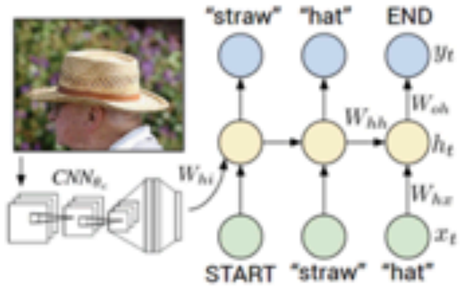
It was very popular a few years ago



Vinyals et al., 2015



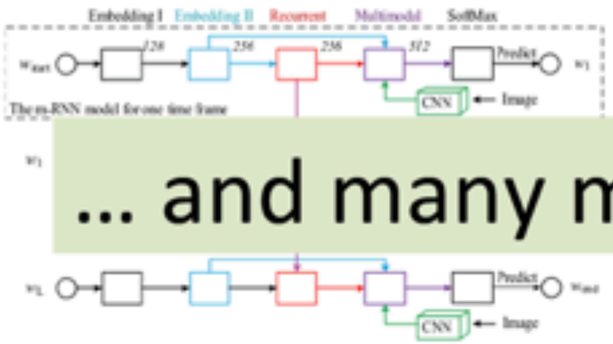
Donahue et al., 2015



Karpathy and Fei-Fei, 2015

Hodosh et al., 2013

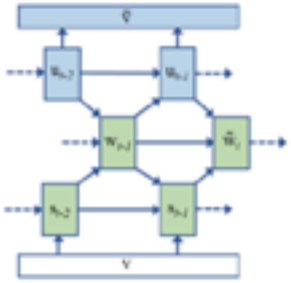
Fang et al., 2015



Mao et al., 2015

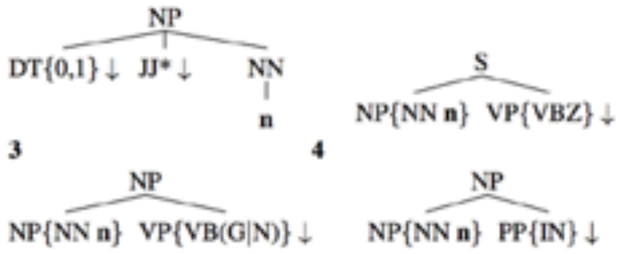
Ordonez et al., 2011

Kulkarni et al., 2011

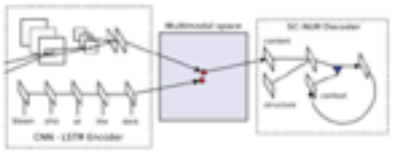


Chen and Zitnick, 2015

Farhadi et al., 2010



Mitchell et al., 2012

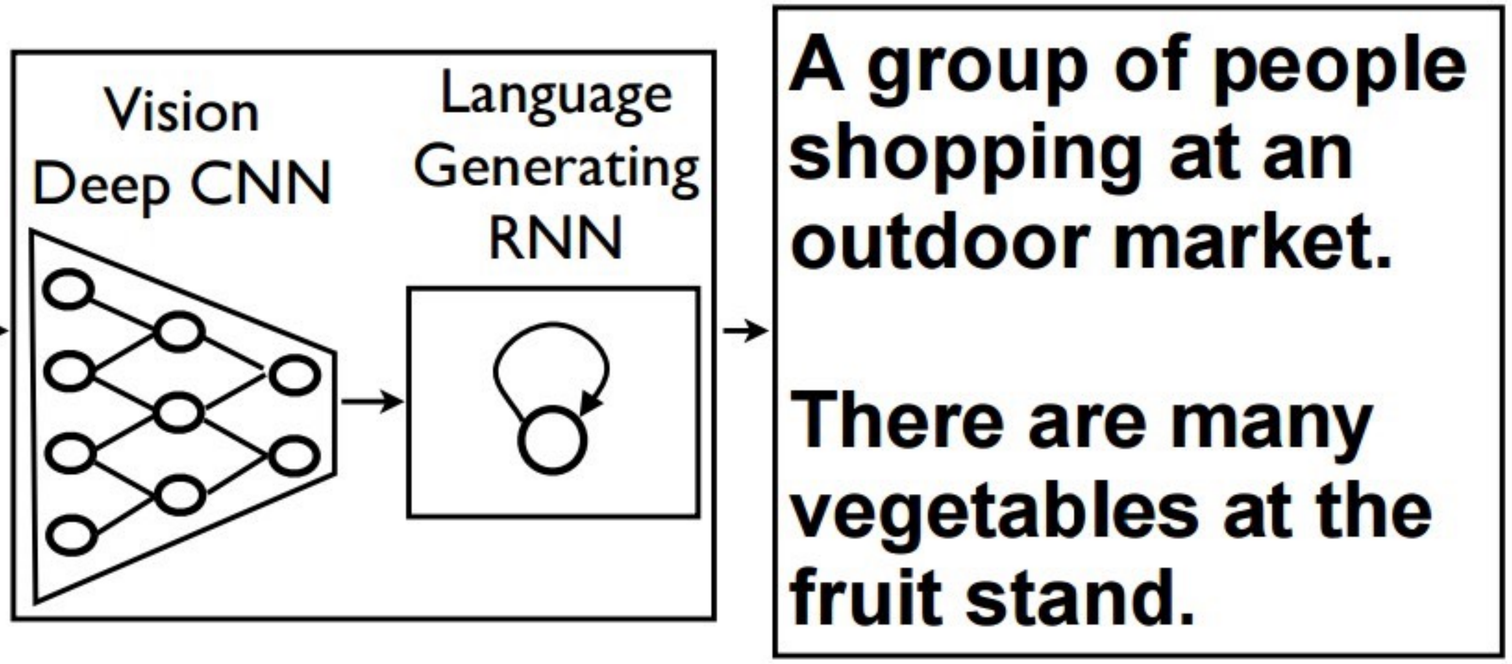


Kiros et al., 2015

Source: D. Parikh

Show and Tell: A Neural Image Caption Generator

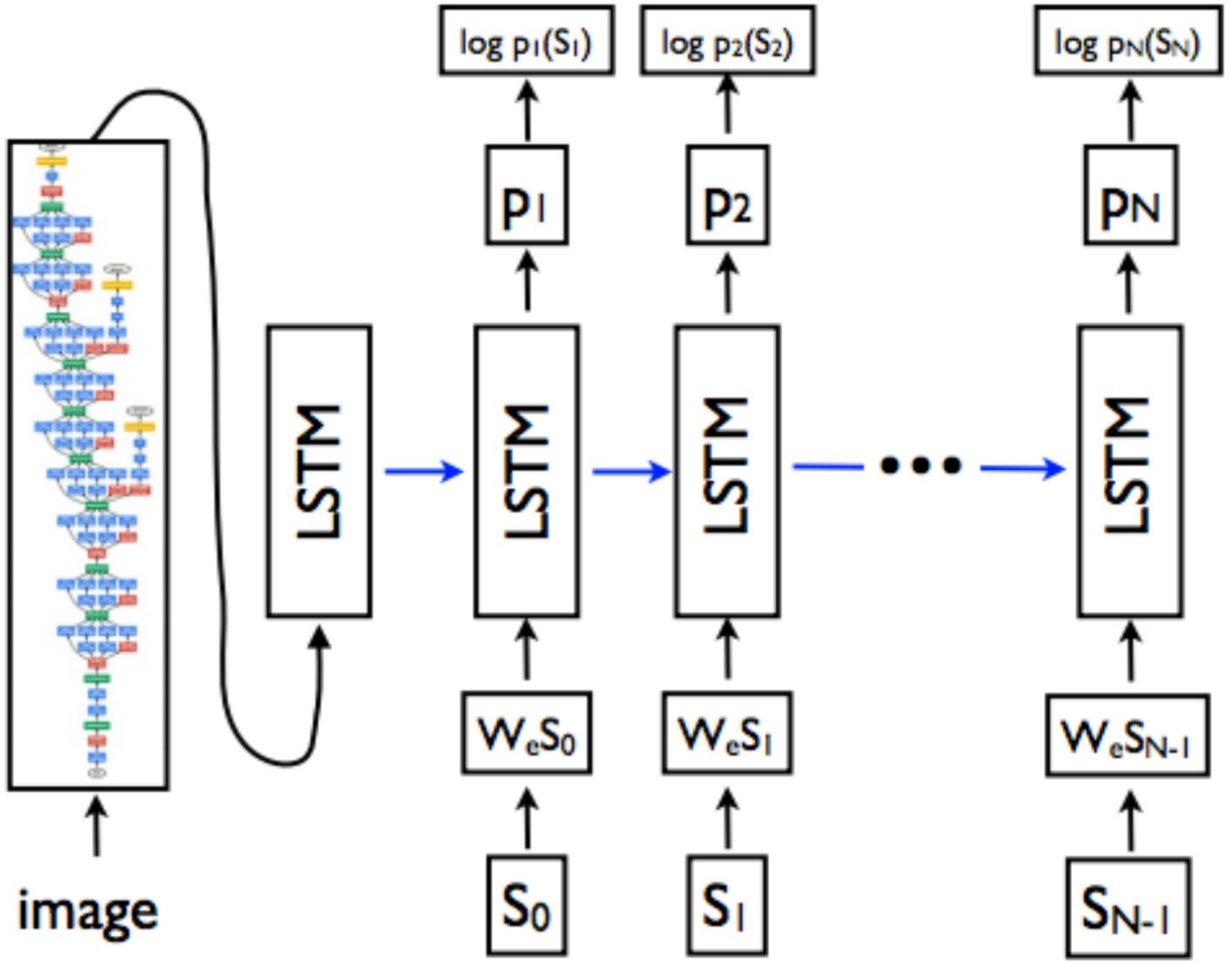
[Vinyals et. al., CVPR 2015]

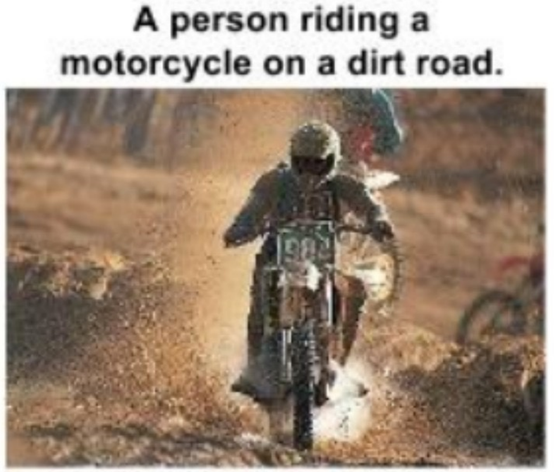


Source: Isola, Torralba, Freeman

Show and Tell: A Neural Image Caption Generator

[Vinyals et. al., CVPR 2015]



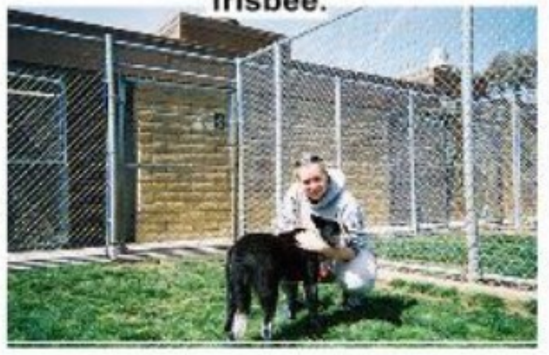


A person riding a motorcycle on a dirt road.

Two dogs play in the grass.

A skateboarder does a trick on a ramp.

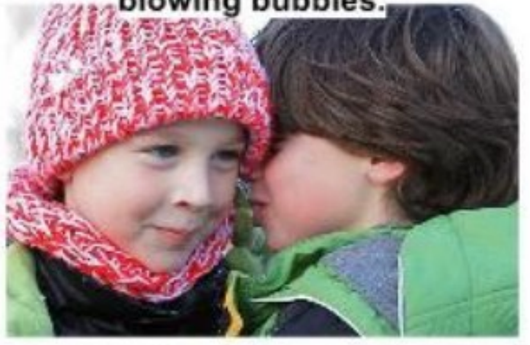
A dog is jumping to catch a frisbee.



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.

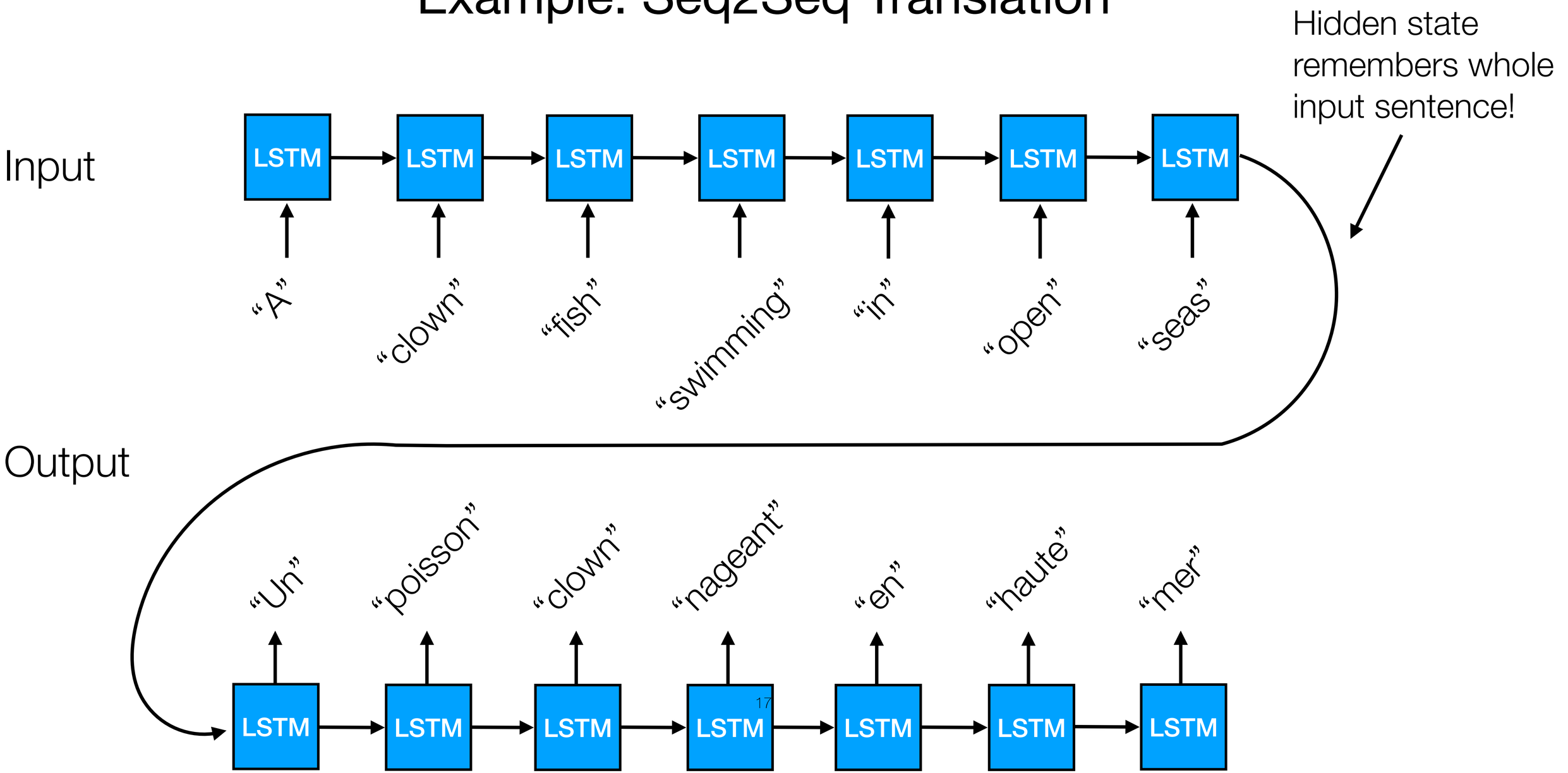
A yellow school bus parked in a parking lot.

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

Shortcomings of recurrent models

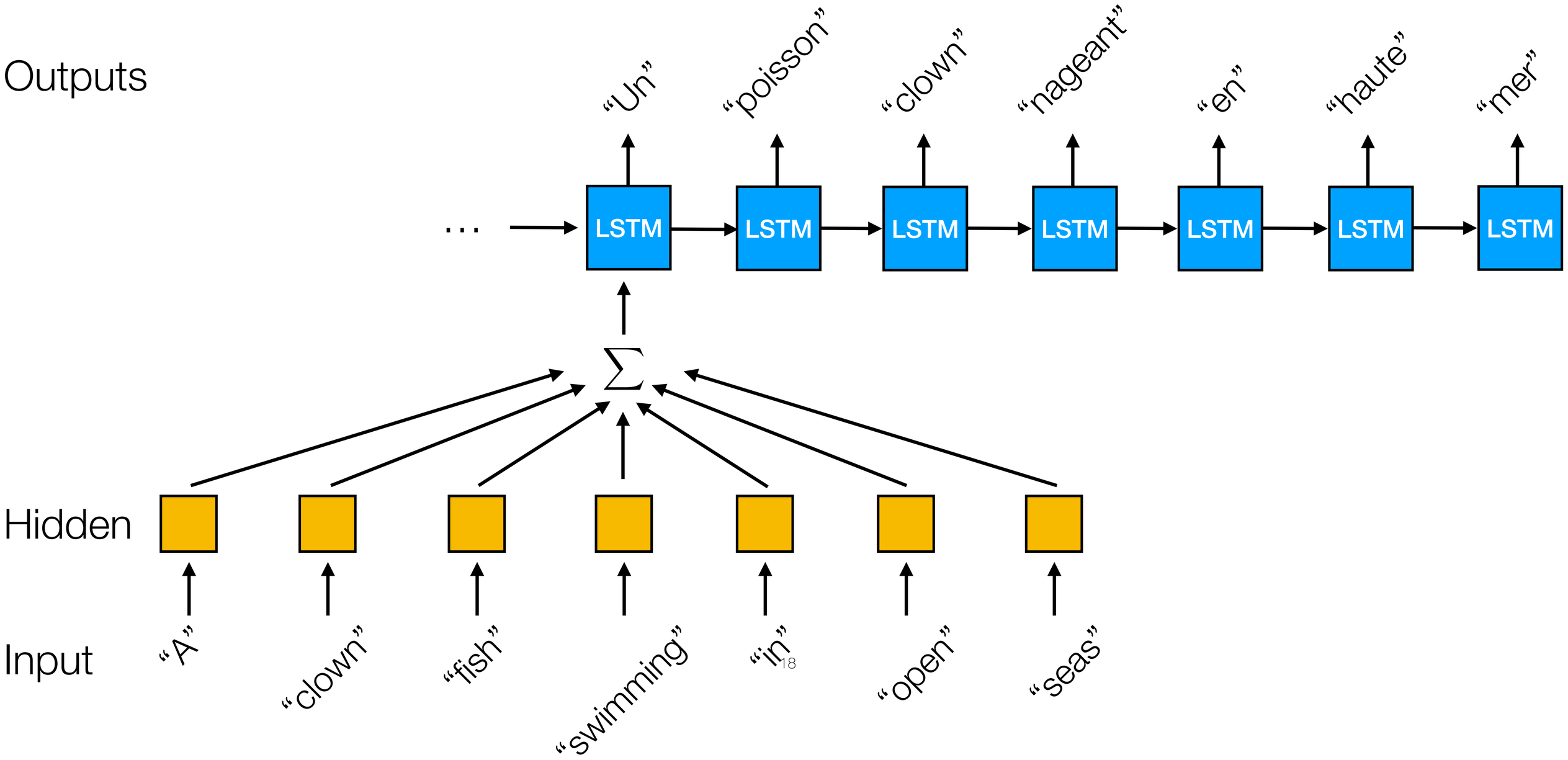
- The recurrent state needs to **remember** a lot
- Instead of remembering: look at the input data! This idea is often implemented using **attention**.
- Example: “sequence to sequence” language translation

Example: Seq2Seq Translation



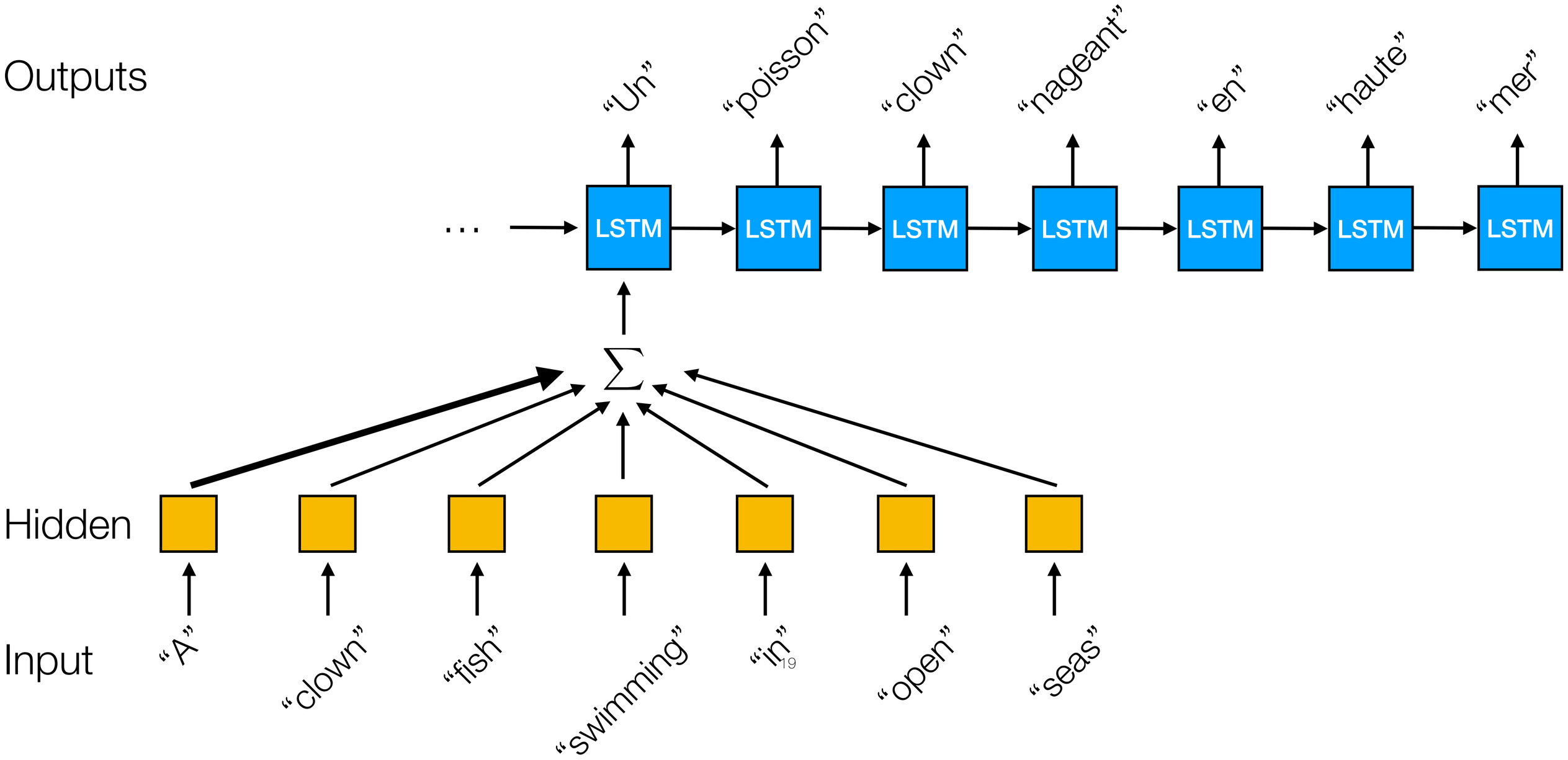
See [Sutskever et al., “Sequence to sequence”, 2014]

Pooling



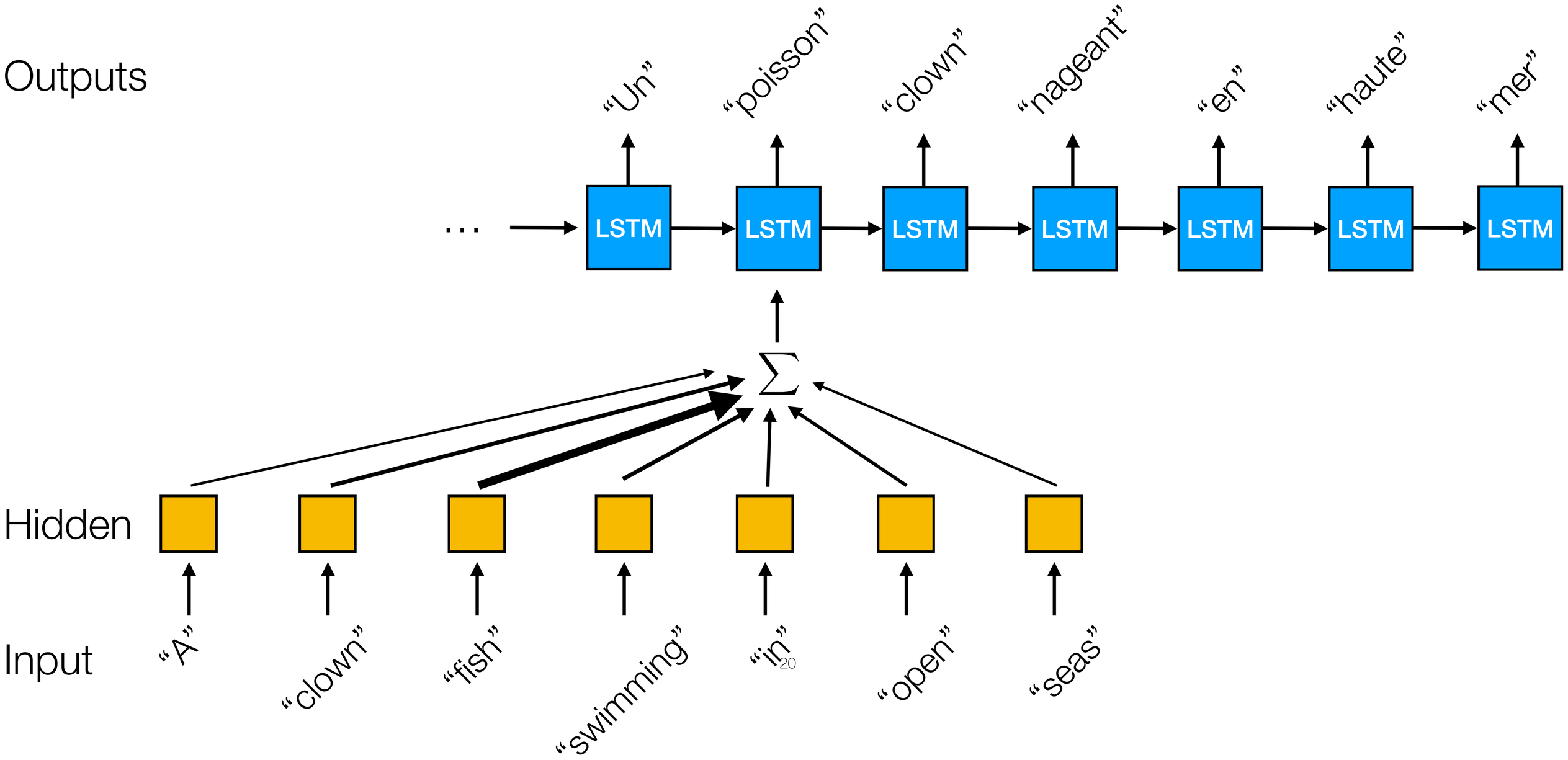
Source: Isola, Torralba, Freeman

Attention



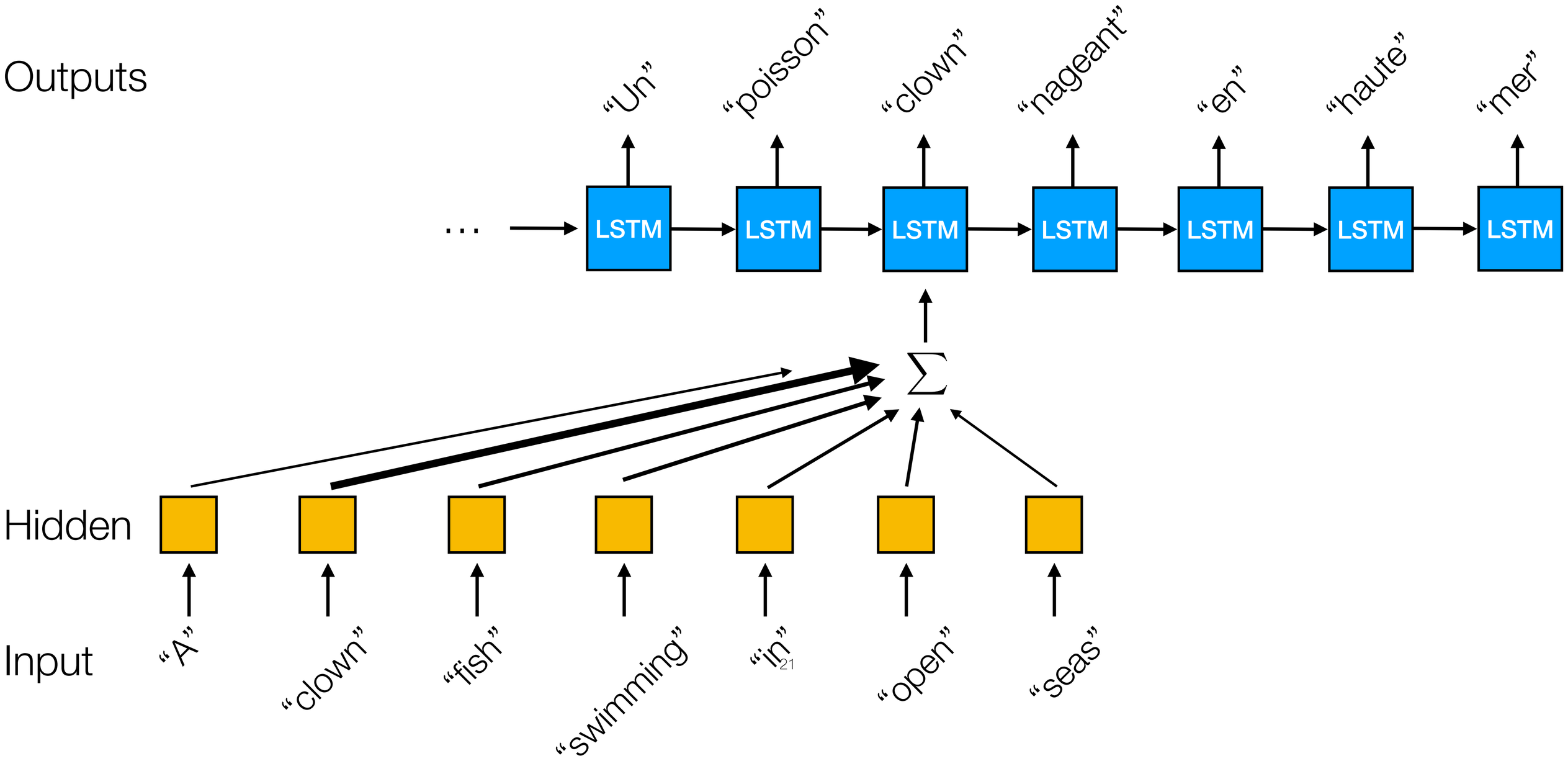
Source: Isola, Torralba, Freeman

Attention



Source: Isola, Torralba, Freeman

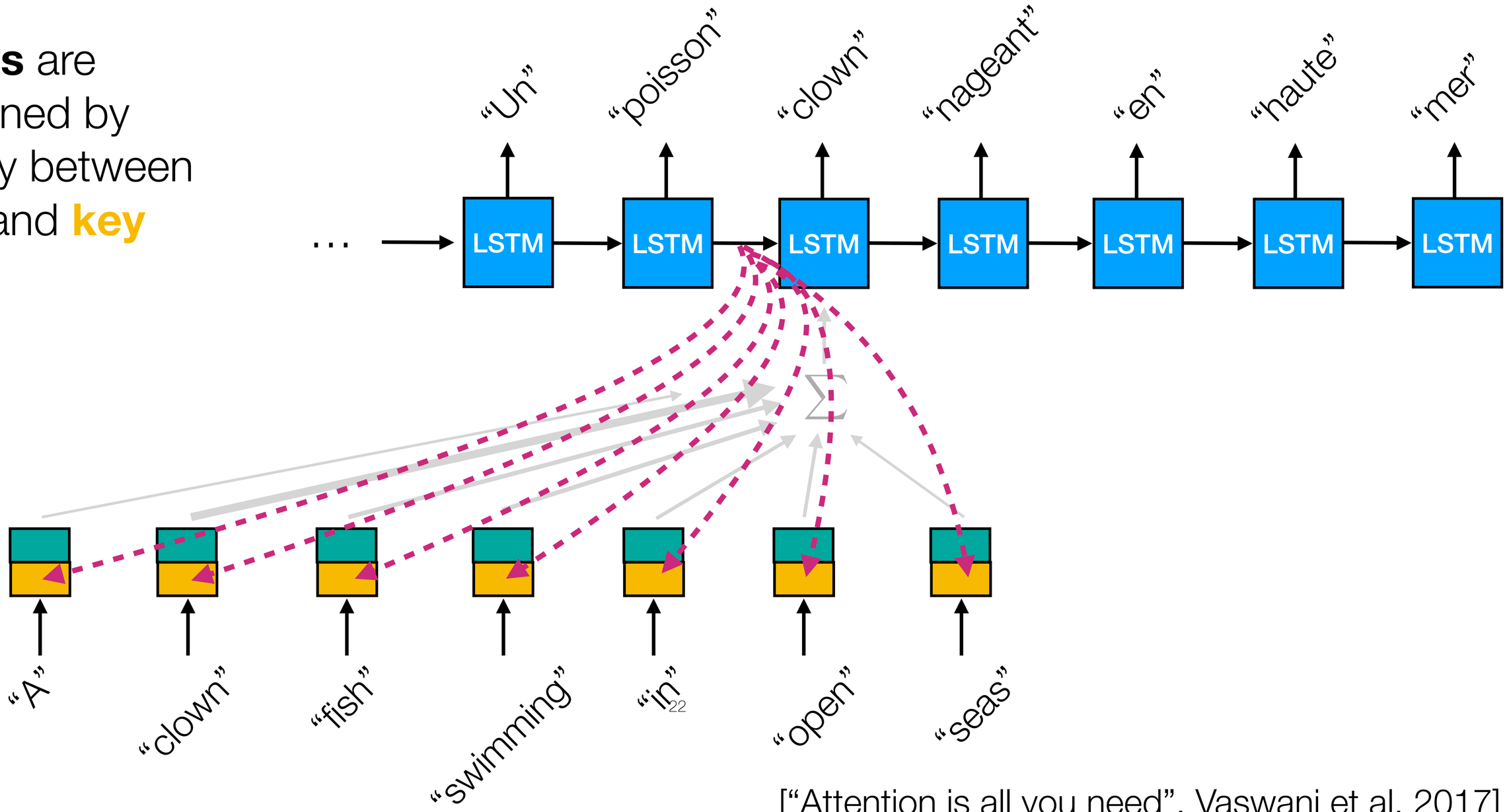
Attention



Source: Isola, Torralba, Freeman

Attention

Weights are determined by similarity between **query** and **key**

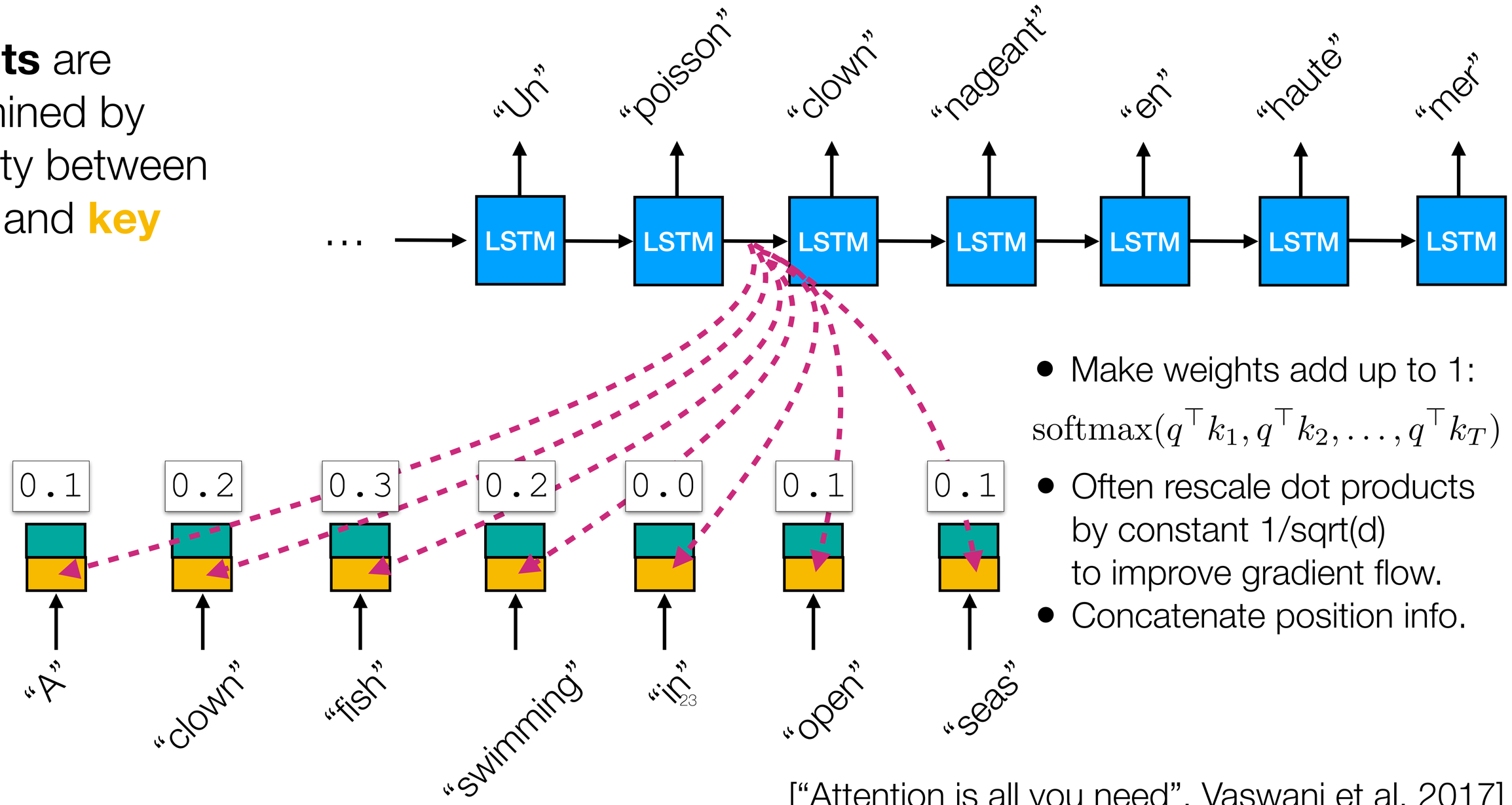


[“Attention is all you need”, Vaswani et al. 2017]

Source: Isola, Torralba, Freeman

Attention

Weights are determined by similarity between **query** and **key**



- Make weights add up to 1:

$$\mathrm{softmax}(q^\top k_1, q^\top k_2, \ldots, q^\top k_T)$$

- Often rescale dot products by constant 1/sqrt(d) to improve gradient flow.
- Concatenate position info.

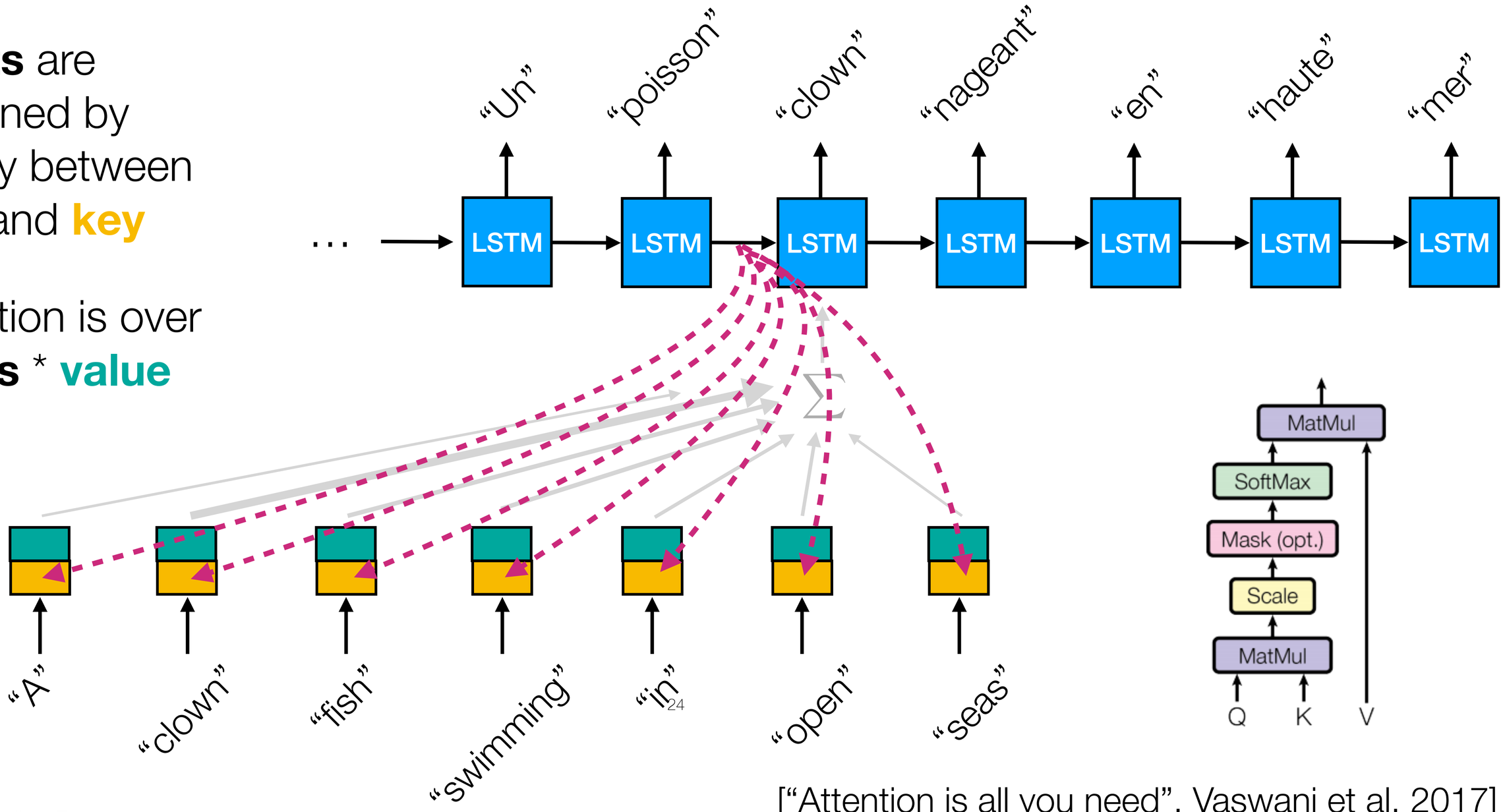
[“Attention is all you need”, Vaswani et al. 2017]

Source: Isola, Torralba, Freeman

Attention

Weights are determined by similarity between **query** and **key**

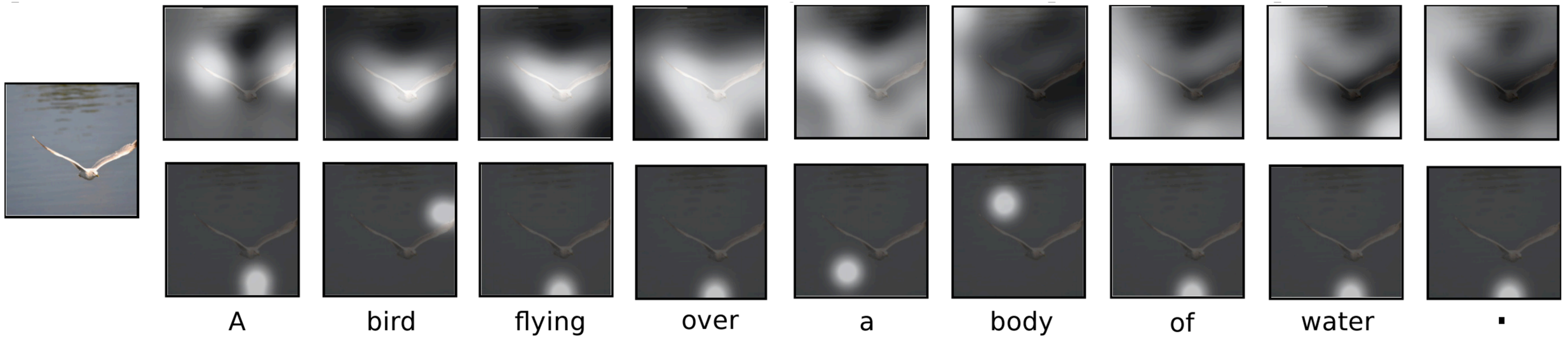
summation is over **weights** * **value**



[“Attention is all you need”, Vaswani et al. 2017]

Source: Isola, Torralba, Freeman

Image captioning with attention



[Xu et al. “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, 2016]