CS273B lecture 4: Recurrent Neural Networks

James Zou

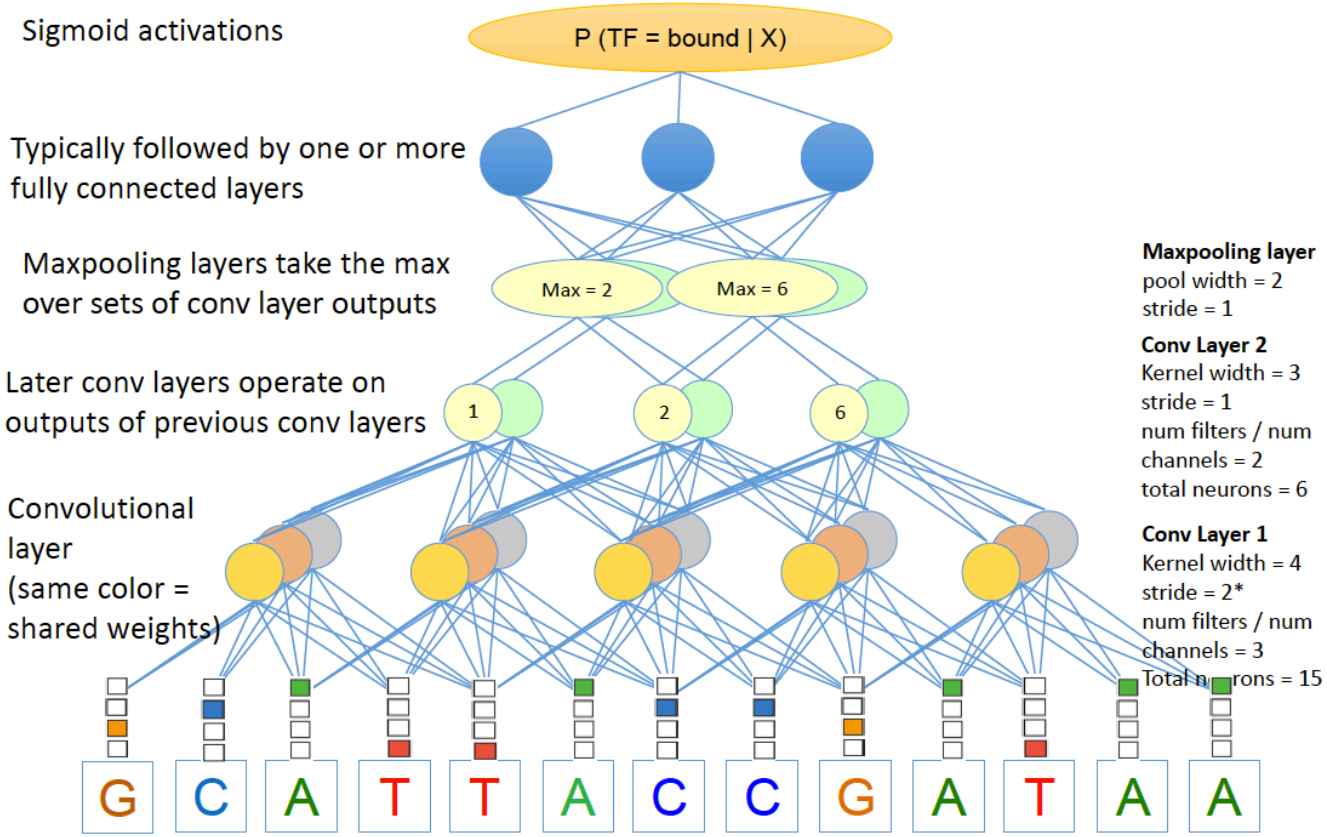
October 4, 2017

Outline

1. Recap of convnets.
2. Simple recurrent neural networks.
3. More complex recurrent architecture: LSTM.
4. Examples: functional genomics and MinION base-calling.

Recap

Deep convolutional neural network



Recap: feedforward and convnets

Main take-aways:

- Composition. Units/layers of a NN are modular and can be composed to form complex architecture.

- Weight-sharing. Enforcing that the weight be equal across a set of units can dramatically decrease # of parameters.

What are limitations of convnets?

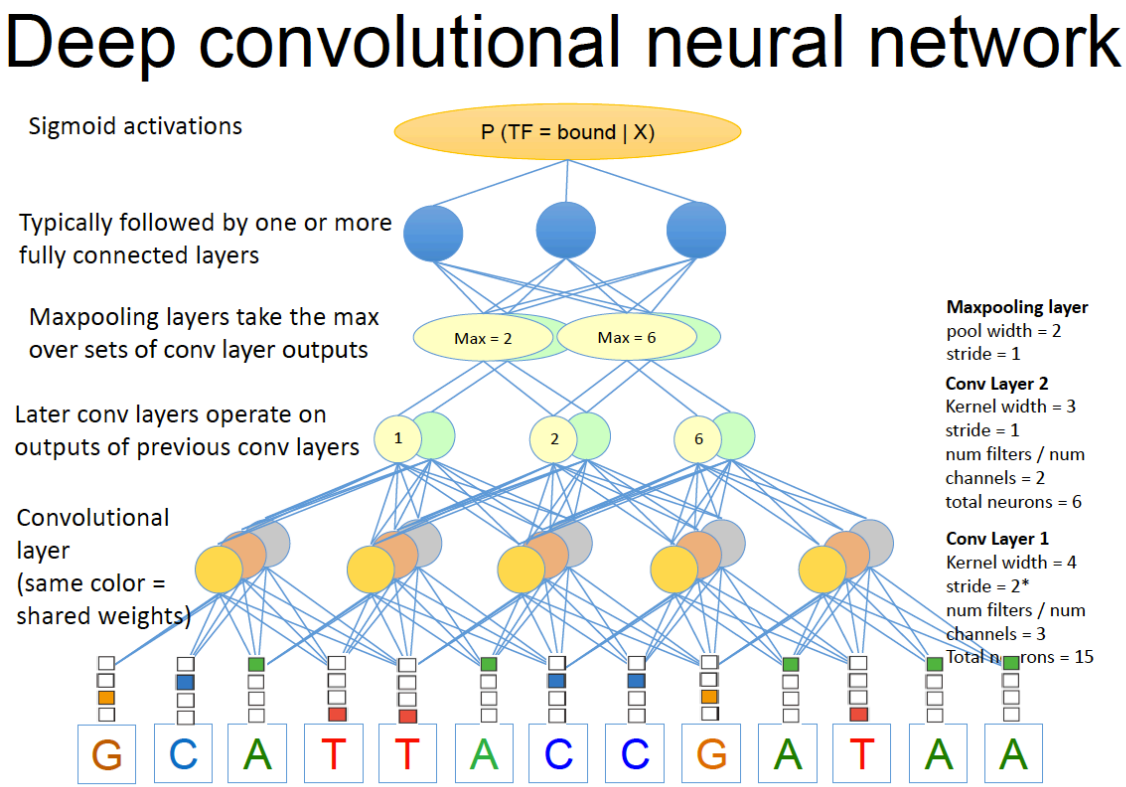
What are limitations of convnets?

- Fixed input length.
- Unclear how to adapt to time-series data.
- Convolution corresponds to strong prior—not appropriate for many biological settings.
- How would CNN capture the grammar?

motif A $\underbrace{\text{-------}}_{\text{variable length} < 100}$ motif B

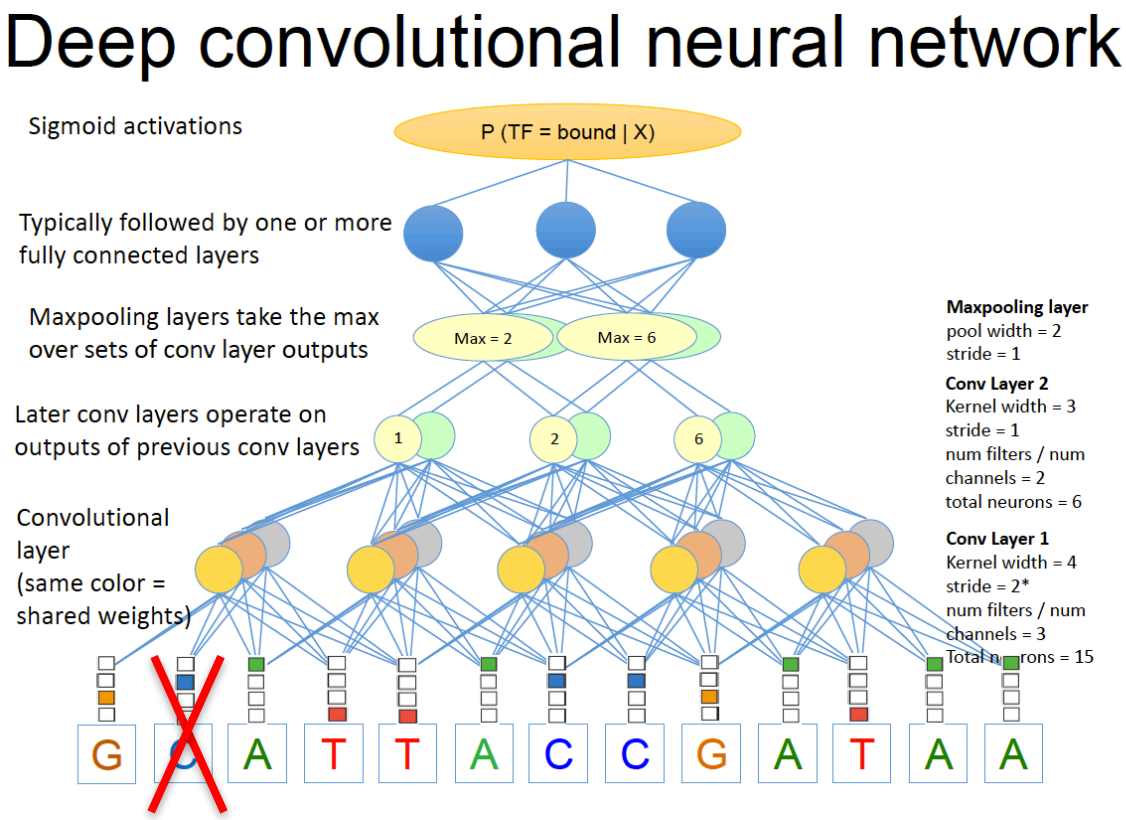
What are limitations of convnets?

- Fixed input length.

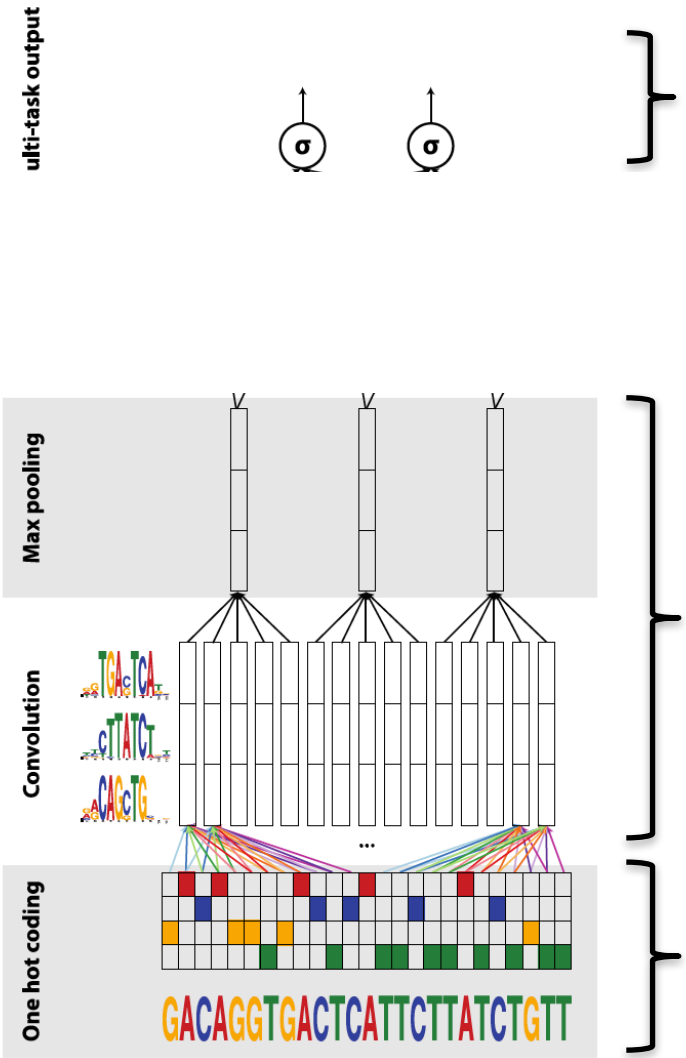


What are limitations of convnets?

- Fixed input length.



LSTM application: enhancer/TF prediction



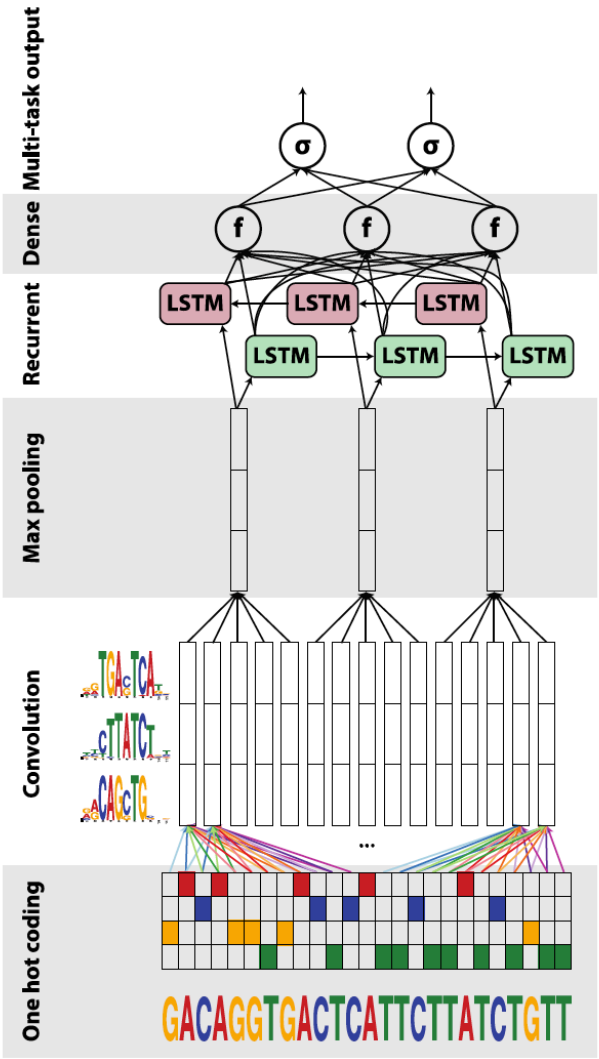
Output: 919 binary vector for the presence of TF/chromatin

Similar convolutional architecture as before

Input: 1kb sequence

Quang and Xie. DanQ. 2016

LSTM application: enhancer/TF prediction



Output: 919 binary vector for the presence of TF/chromatin

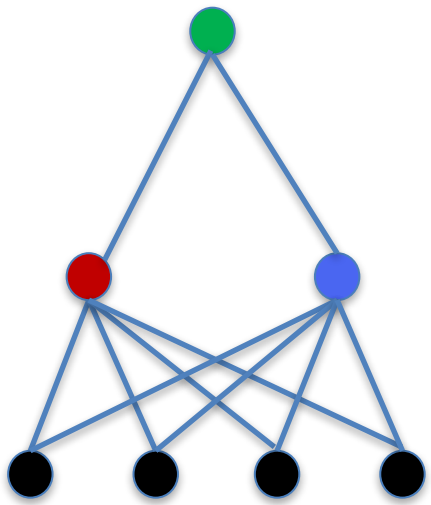
Bi-directional LSTM

Similar convolutional architecture as before

Input: 200bp sequence

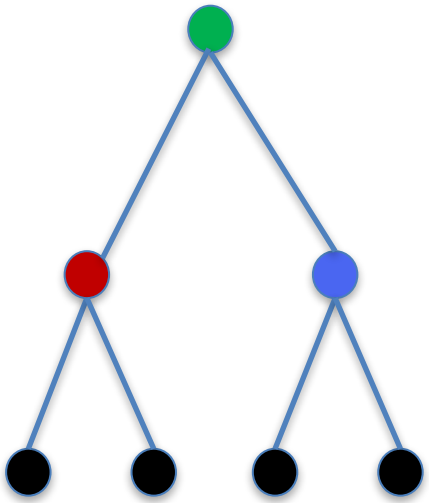
Quang and Xie. DanQ. 2016

Feedforward NN



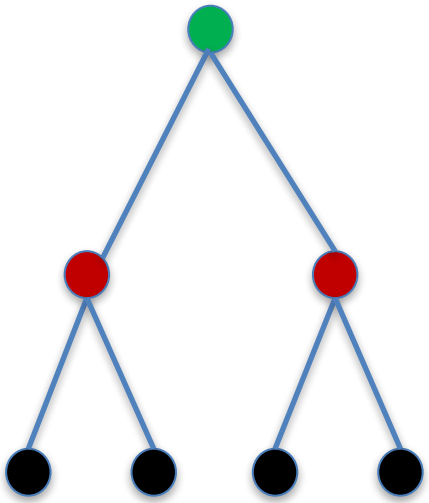
Dense network: every node connected to every other node in the next layer. All different weights.

ConvNet



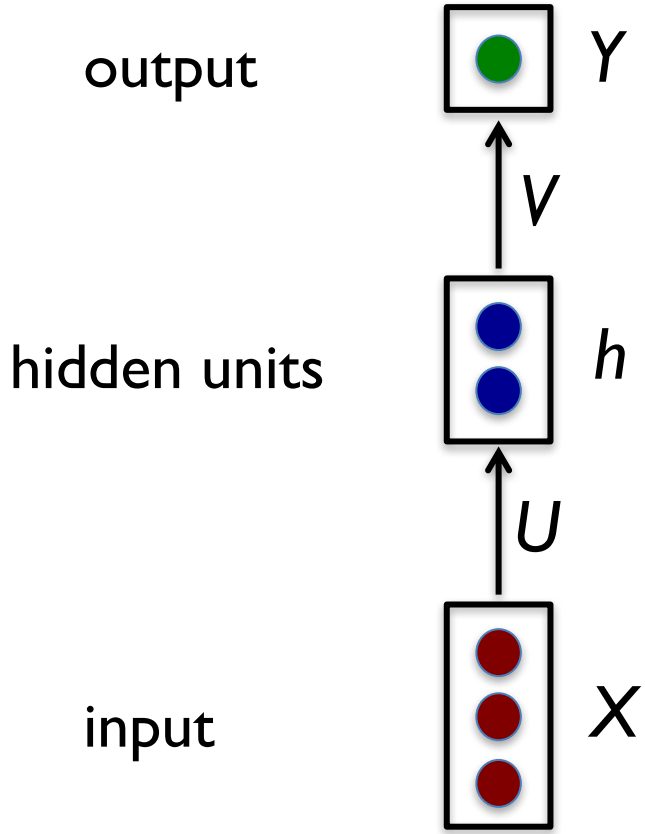
To get to a CNN: we prune some edges and constrain the weights.

ConvNet

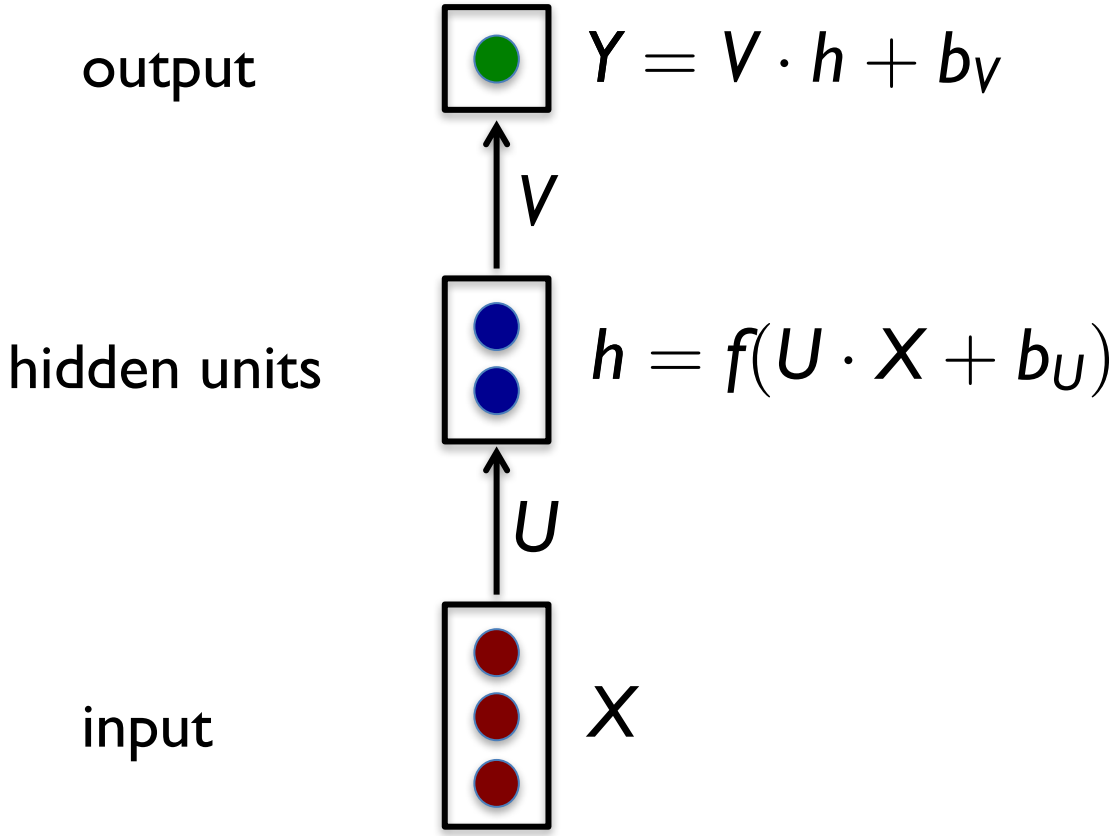


To get to a CNN: we prune some edges and constrain the weights.

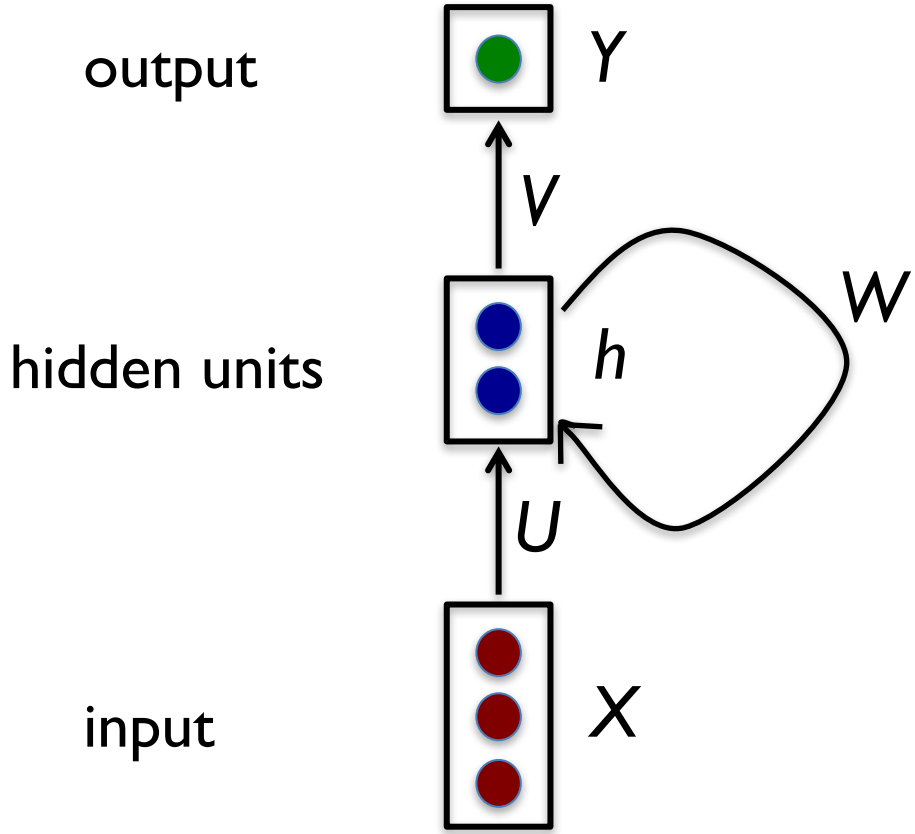
Recurrent neural network



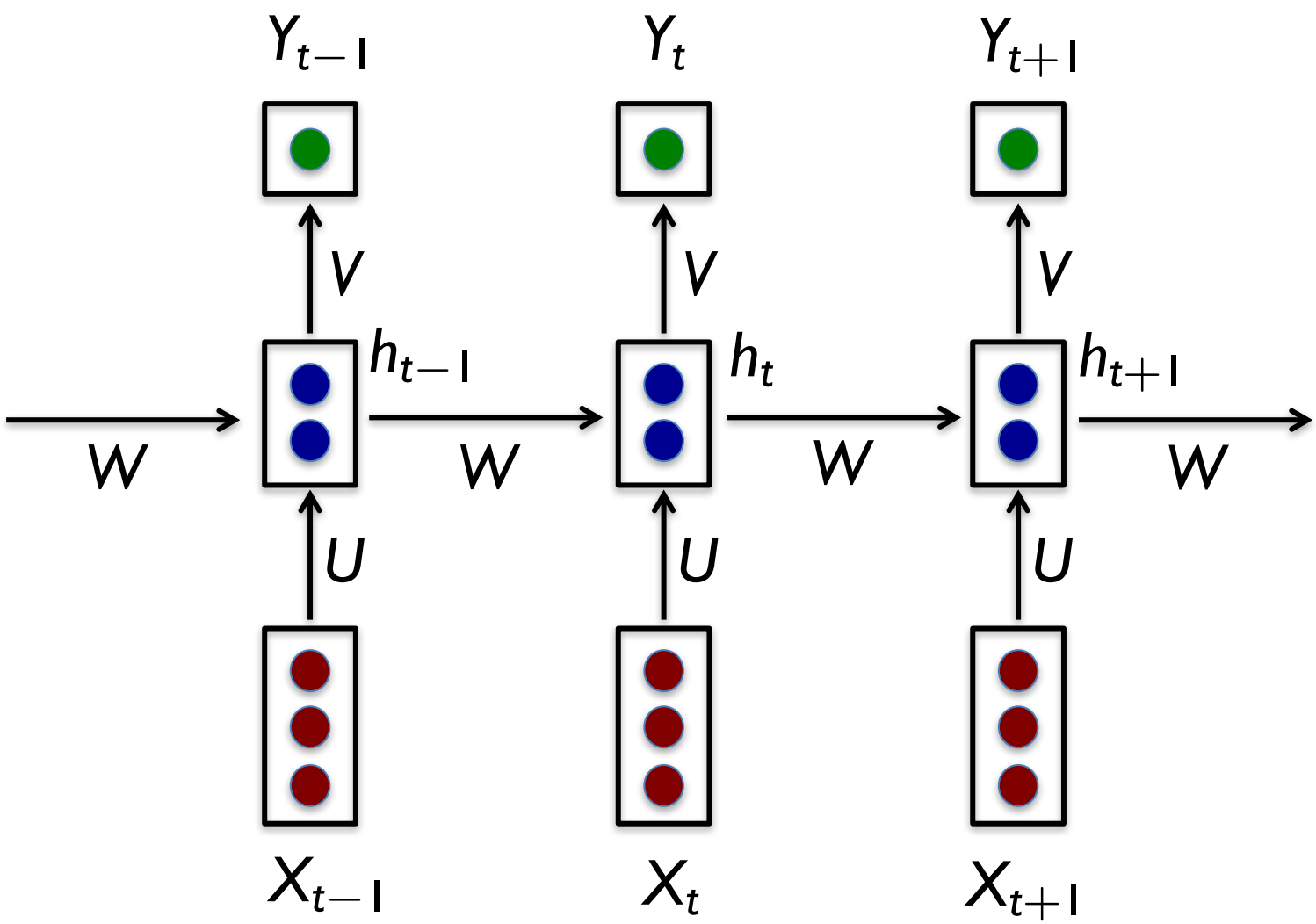
Recurrent neural network



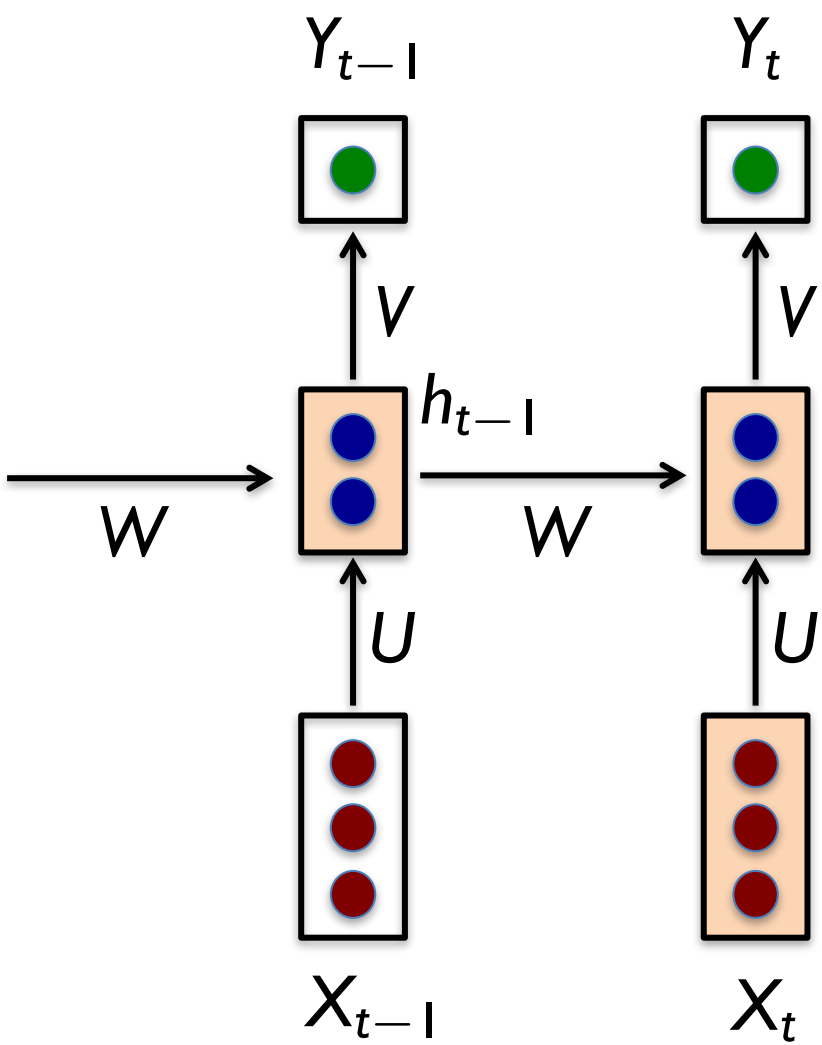
Recurrent neural network



Recurrent neural network

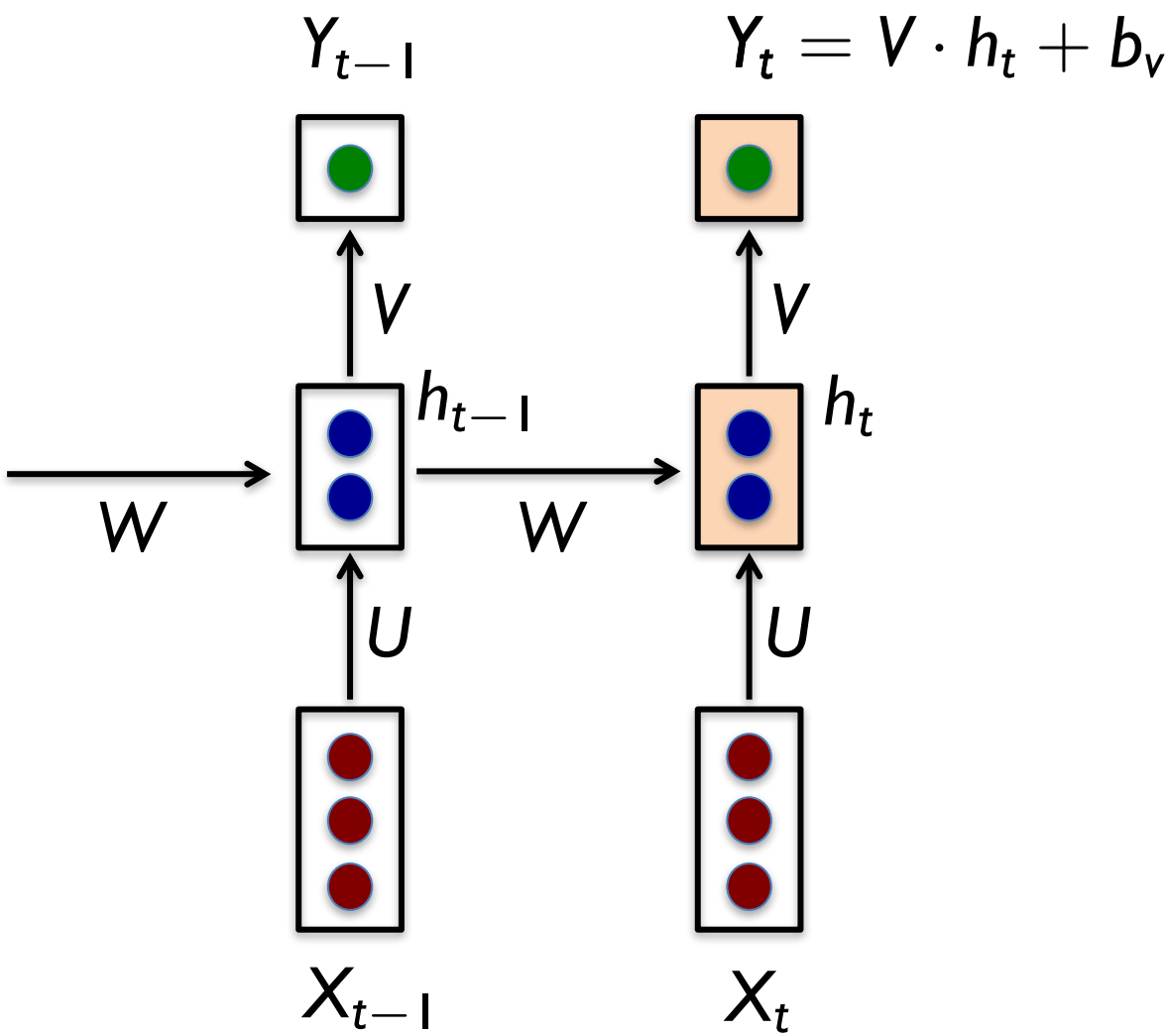


Recurrent neural network

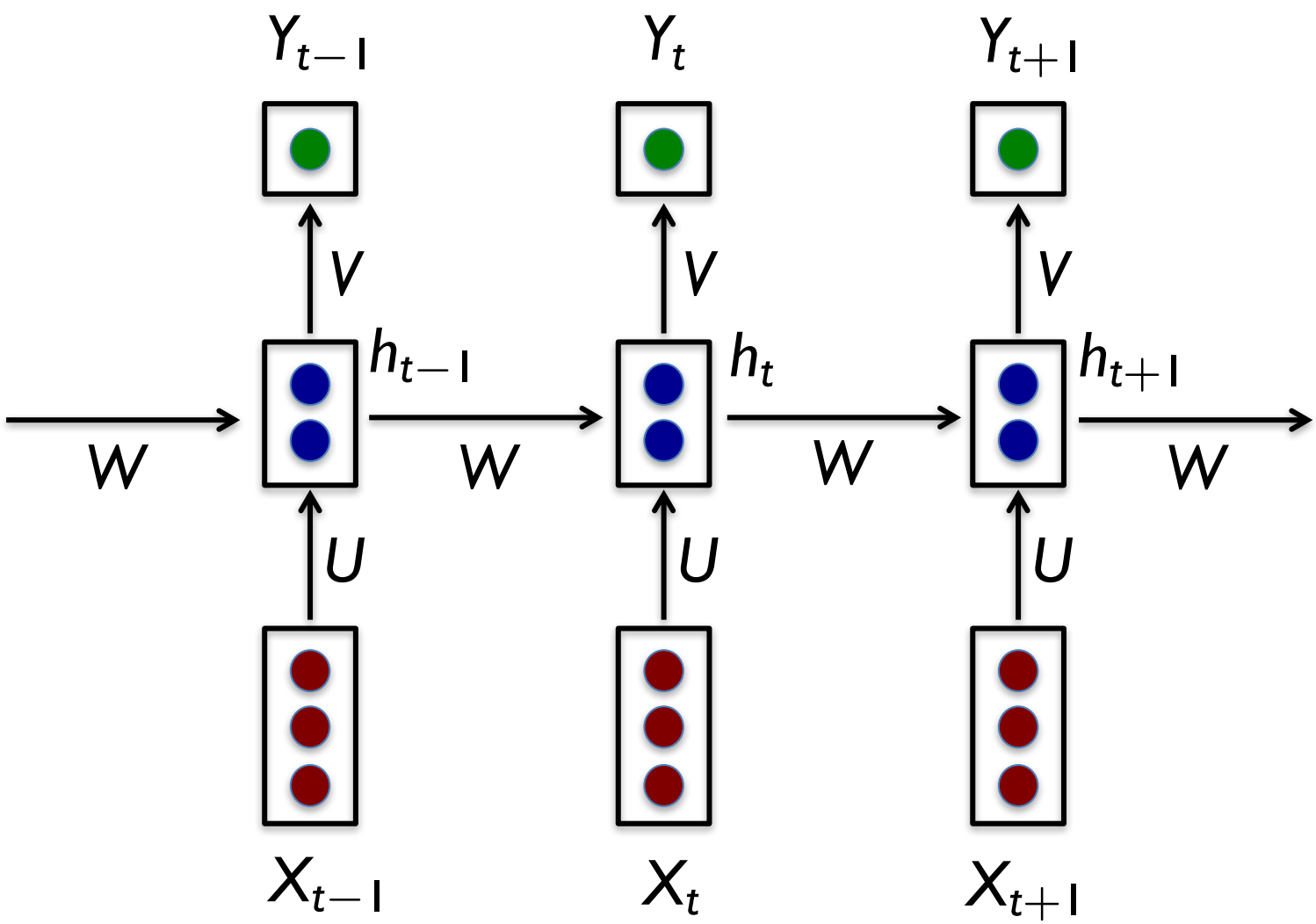


$$h_t = f(U \cdot X_t + W \cdot h_{t-1} + b)$$

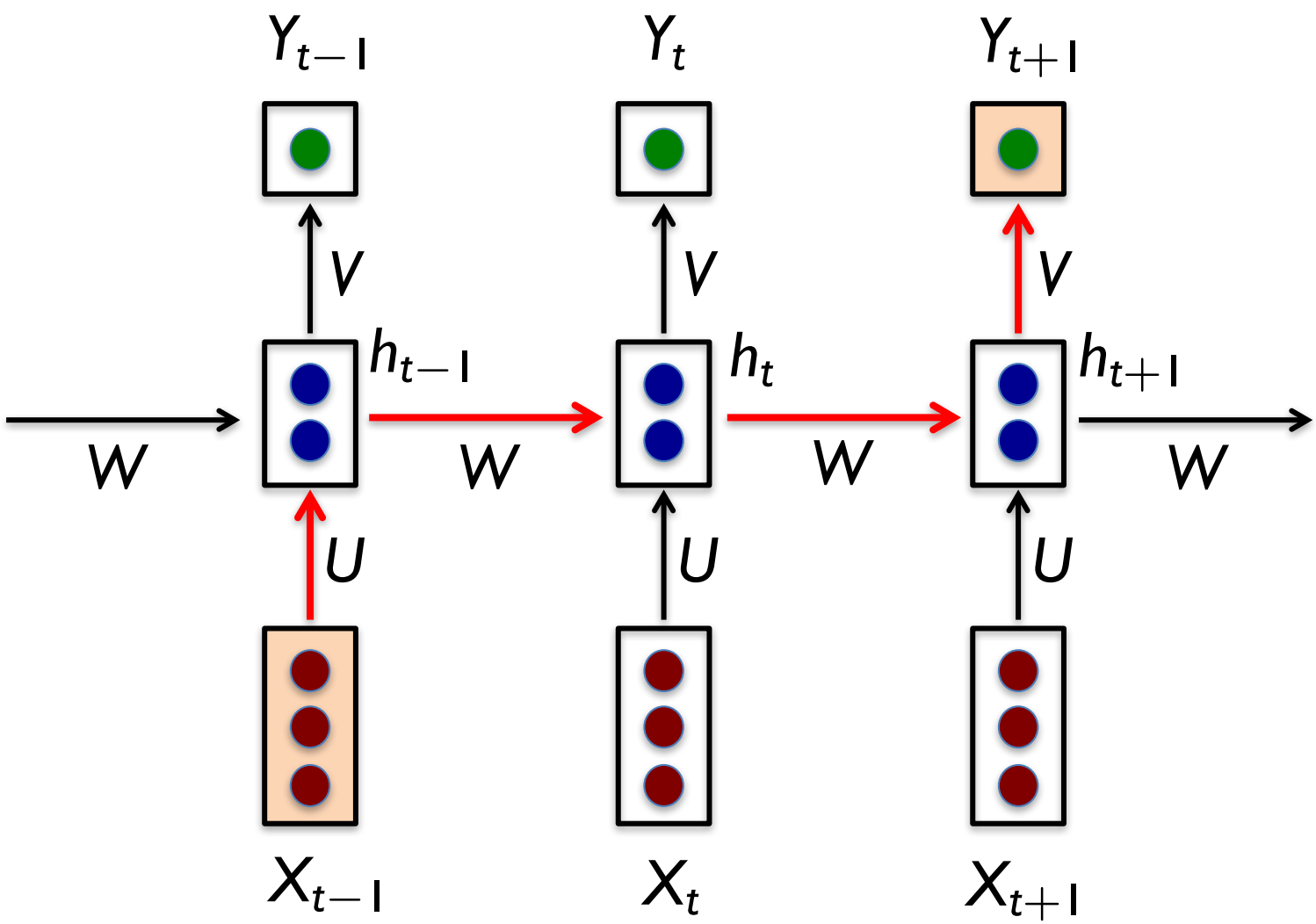
Recurrent neural network



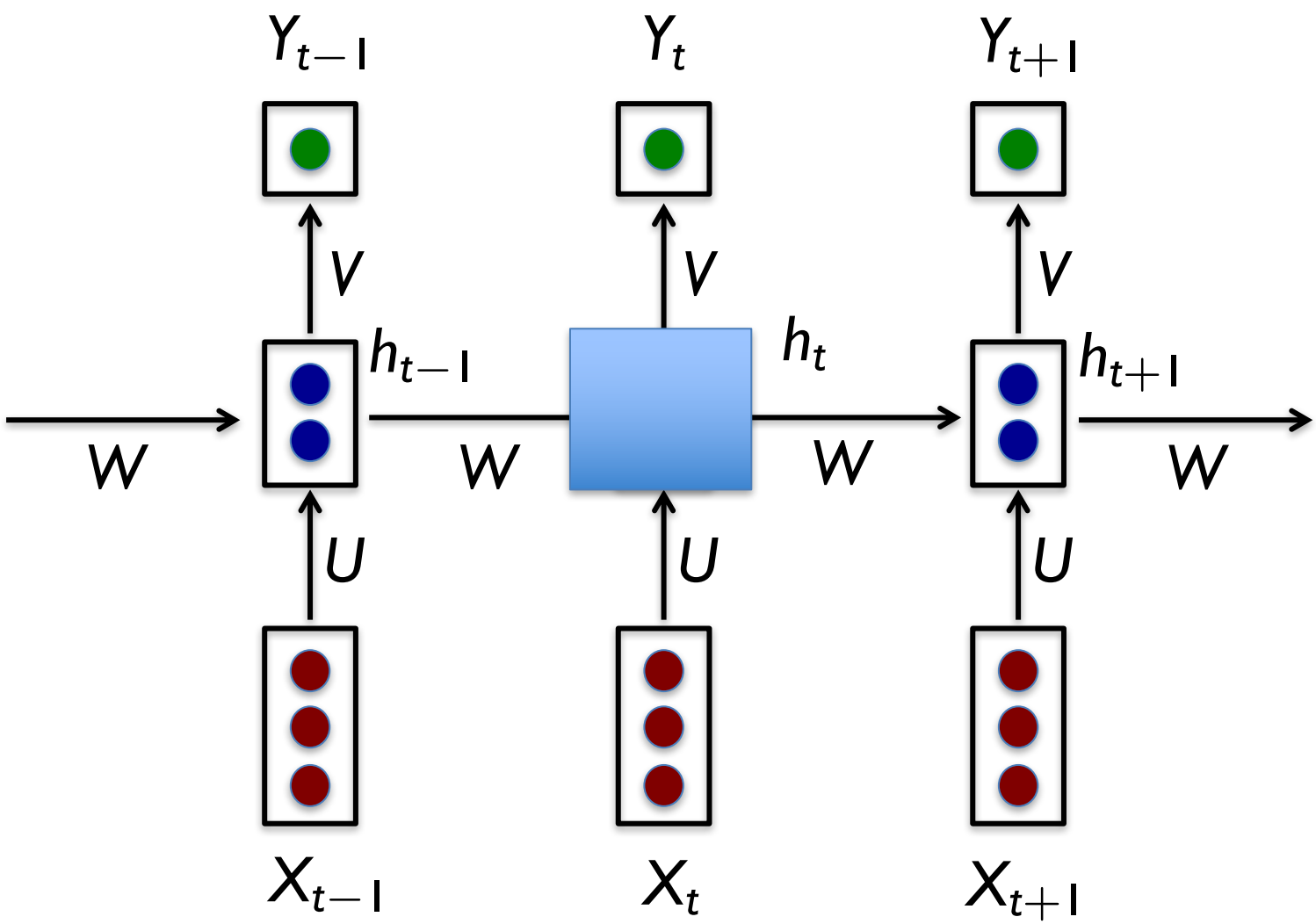
What does RNN remind you of?



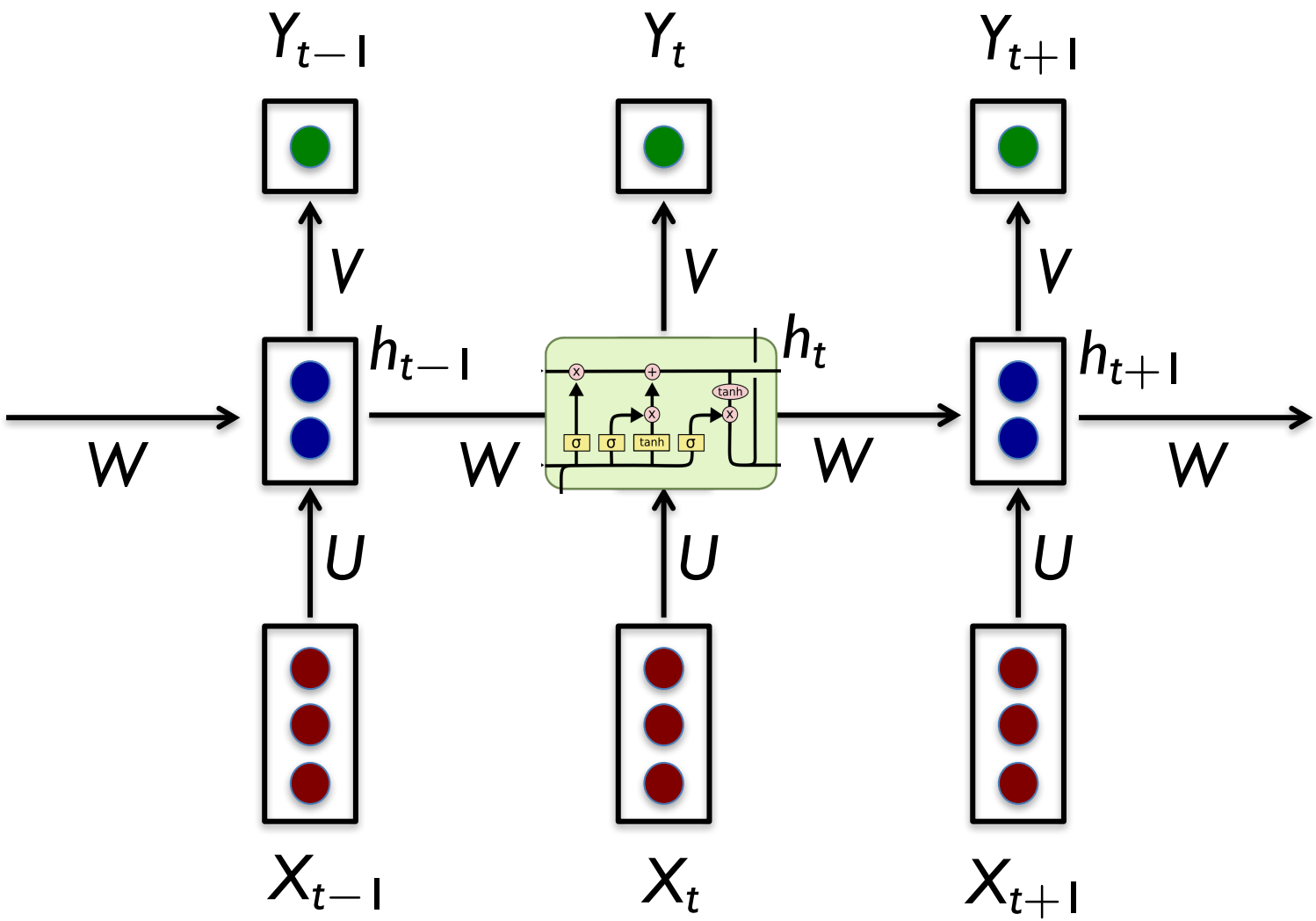
Vanilla RNN: lacks long term memory



More complex architectures



Long short term memory (LSTM)



Hochreiter, Schmidhuber 1997

LSTM: inside the hood

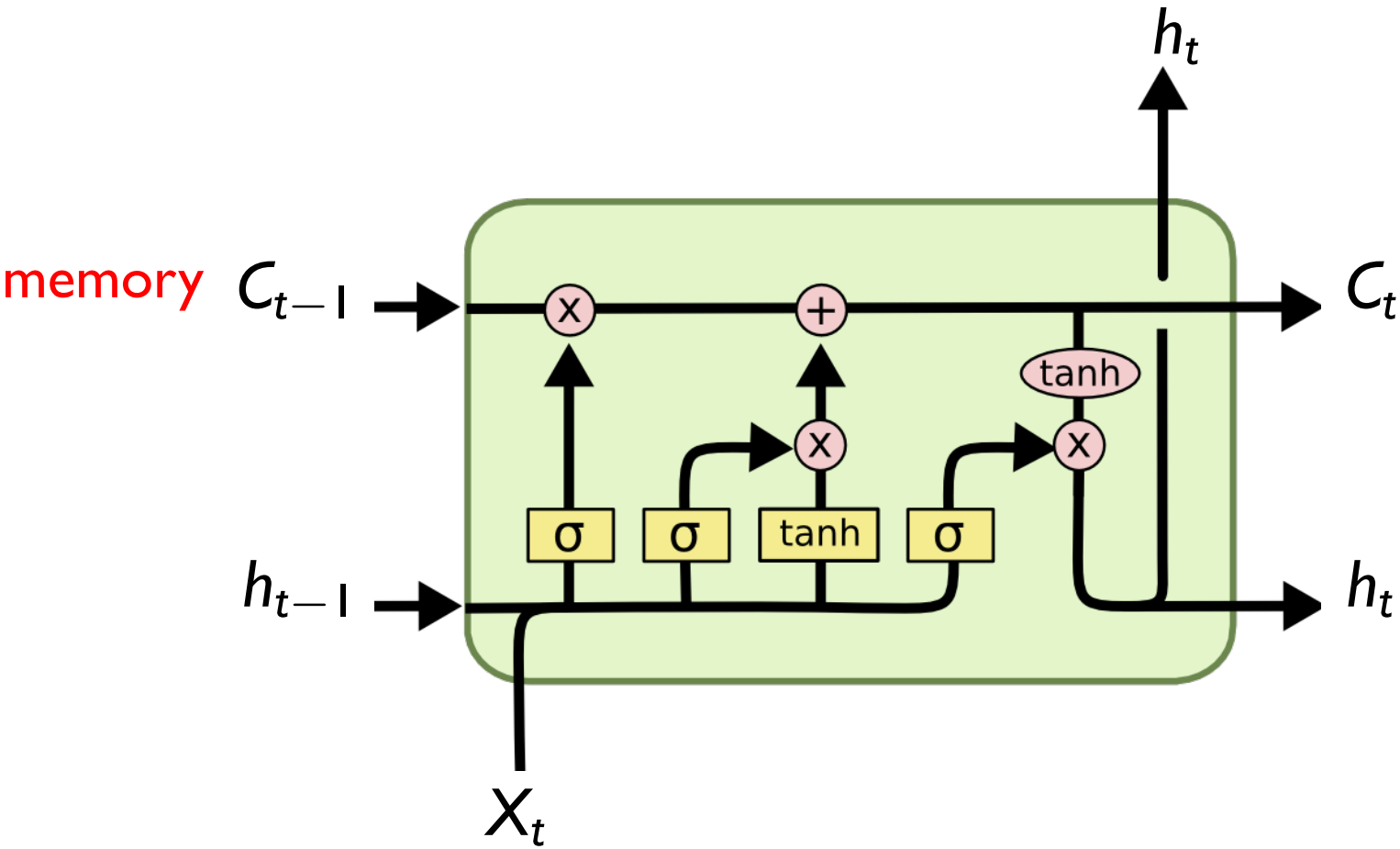


Figure adapted from Olah blog.

LSTM: inside the hood

$$C_t = \boxed{C_{t-1} * \text{forget}} + \text{new memory}$$

$$\text{forget} = \sigma(W_f \cdot [h_{t-1}, X_t] + b_f)$$

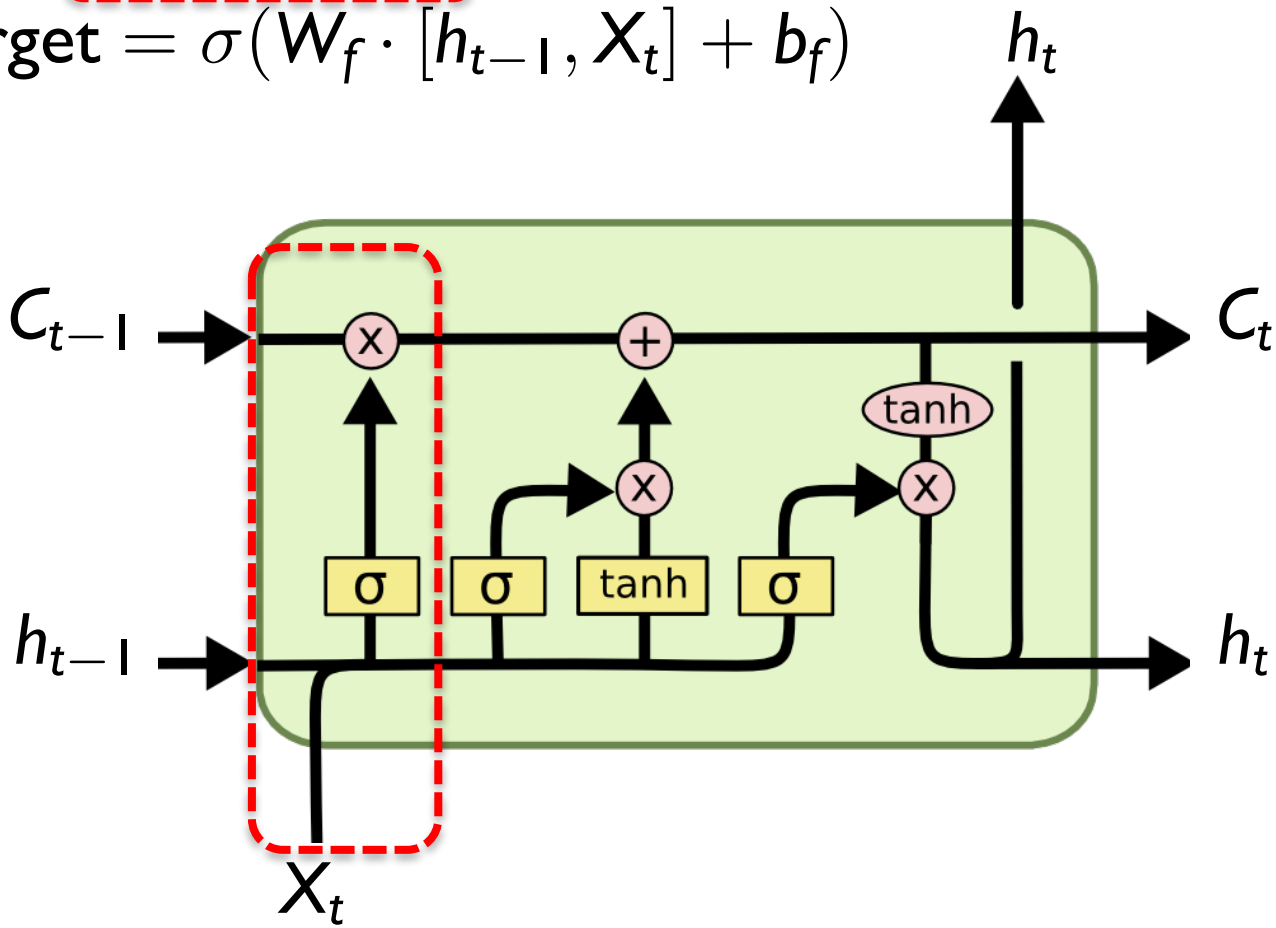


Figure adapted from Olah blog.

LSTM: inside the hood

$$C_t = C_{t-1} * \text{forget} + \boxed{\text{new} * i_t}$$

$$\text{new} = \tanh(W_C \cdot [h_{t-1}, X_t] + b_C)$$

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

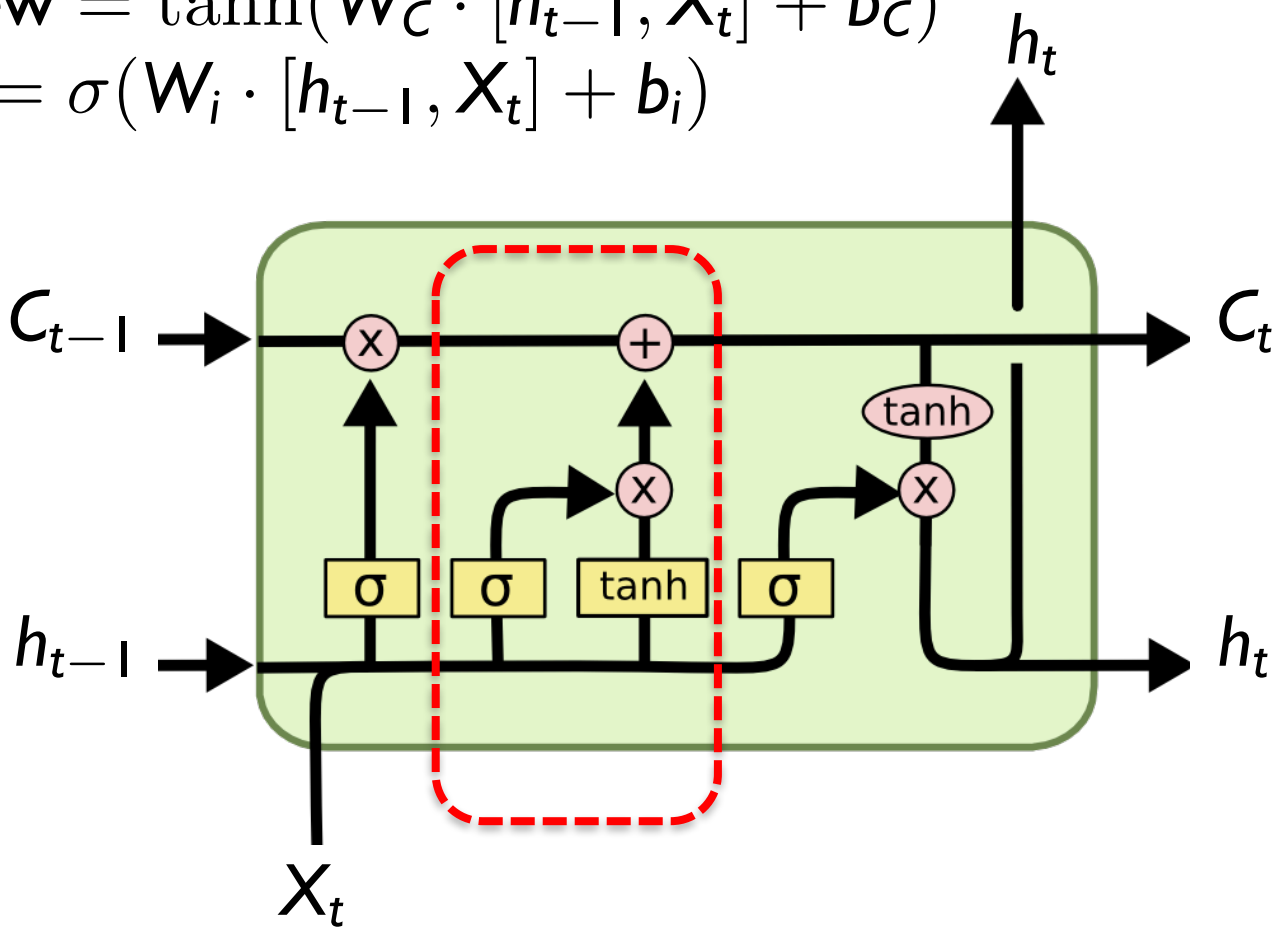


Figure adapted from Olah blog.

LSTM: inside the hood

$$O_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * \tanh(C_t)$$ output

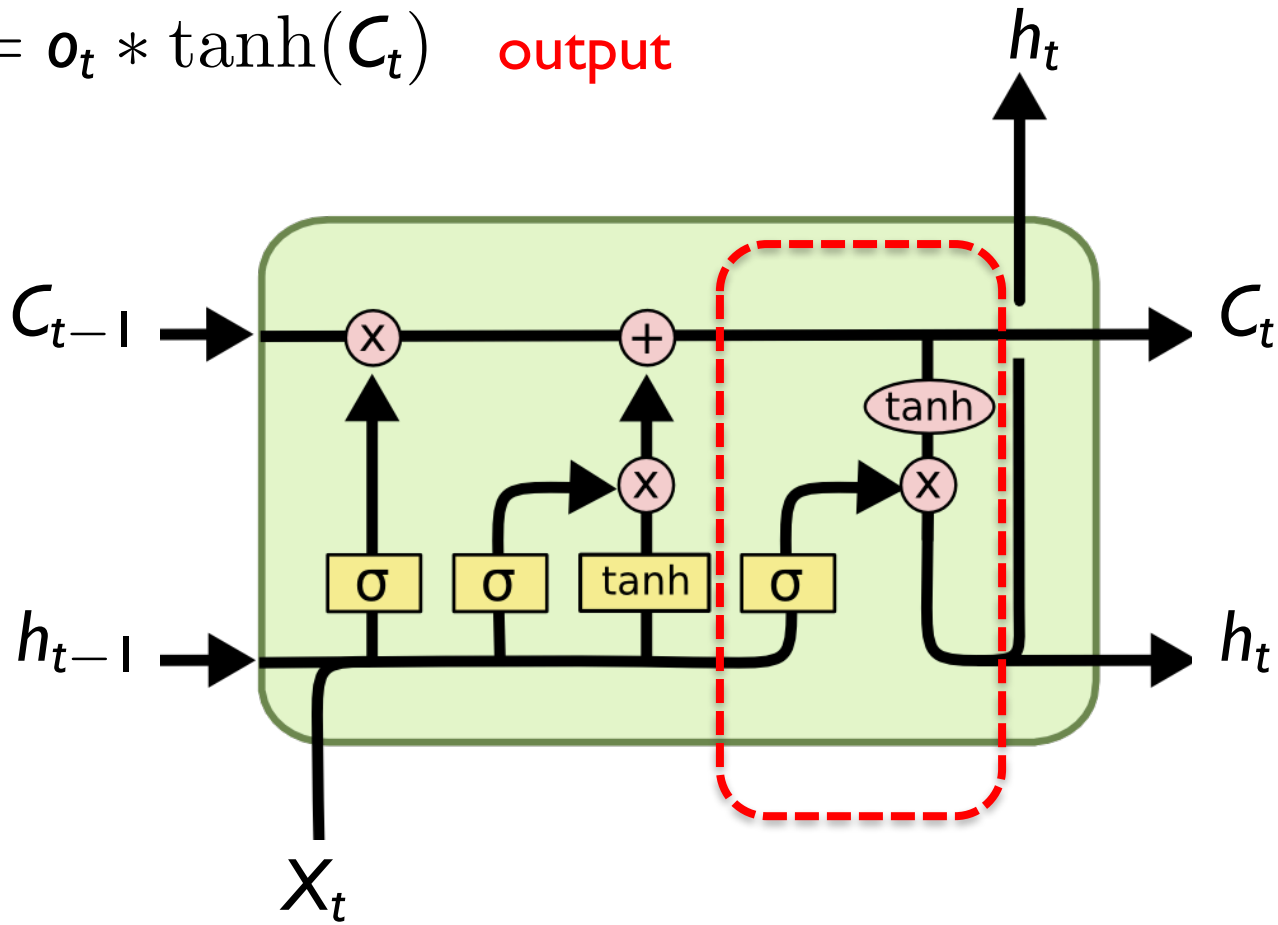


Figure adapted from Olah blog.

LSTM summary

- LSTM is a variant of RNN that makes it easier to retain long-range interactions.

- Parameters of LSTM:

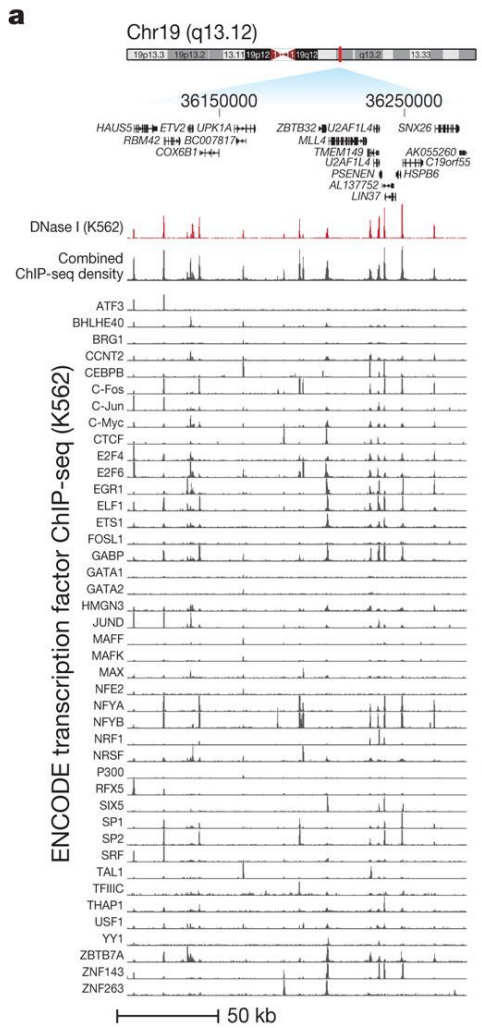
  W_f, b_f forget

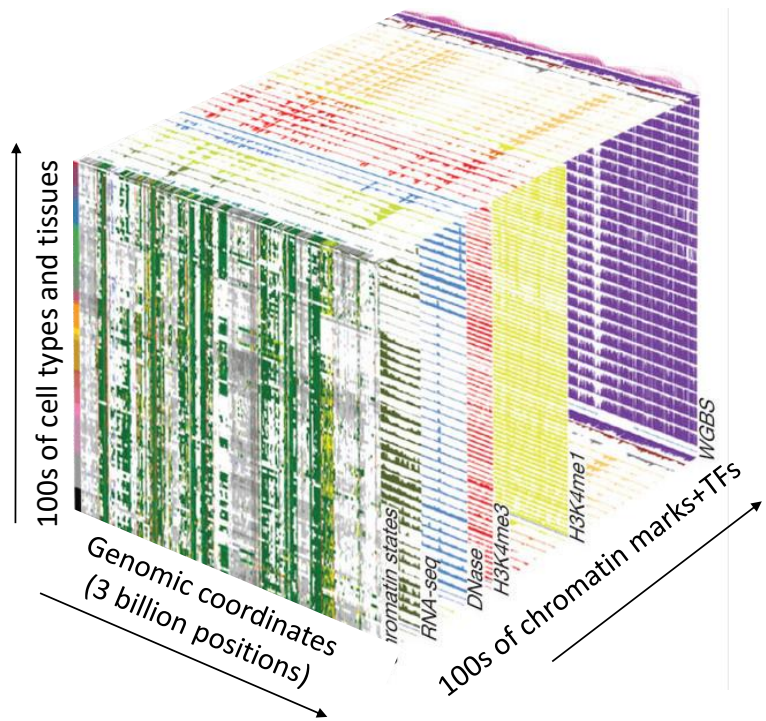
  W_C, b_C new memory

  W_i, b_i weight of new memory (input)

  W_o, b_o output

LSTM application: enhancer/TF prediction





LSTM application: enhancer/TF prediction

919 binary catetories from TF and histone ChIP-seq

0 0 1 0 0 1 1 0 0 0 0 0

ATTCGAAA…TAATTCAA

1kb DNA sequence

Training data:

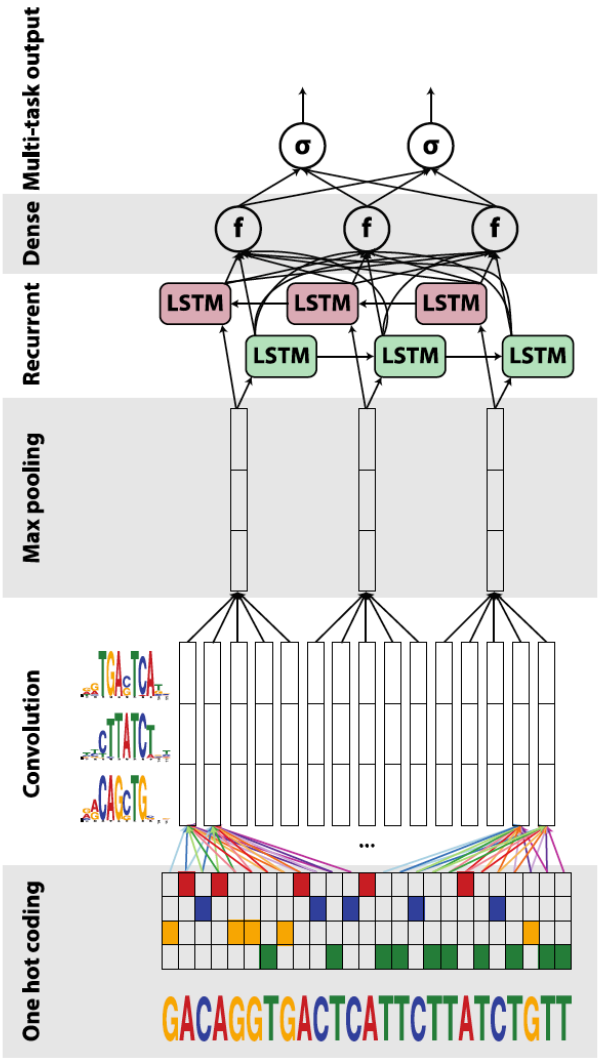
- 1 million labeled 1kb sequences.

Test data:

- Held-out section of a chromosome.

Quang and Xie. DanQ. 2016

LSTM application: enhancer/TF prediction



Output: 919 binary vector for the presence of TF/chromatin

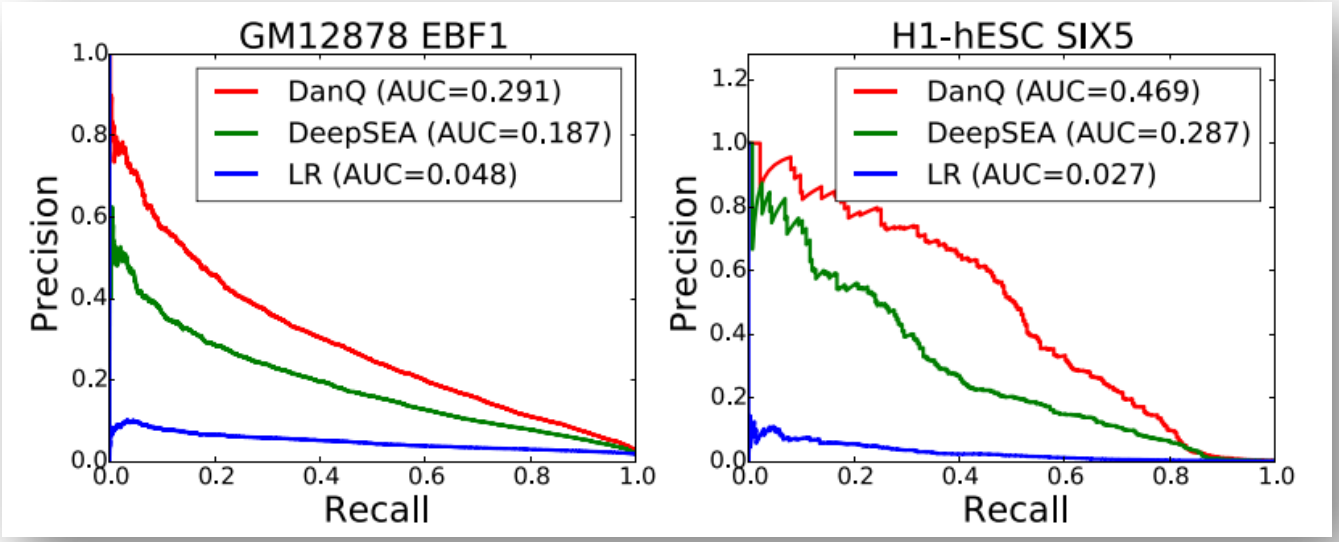
Bi-directional LSTM

Similar convolutional architecture as before

Input: 1kb sequence

Quang and Xie. DanQ. 2016

LSTM application: enhancer/TF prediction



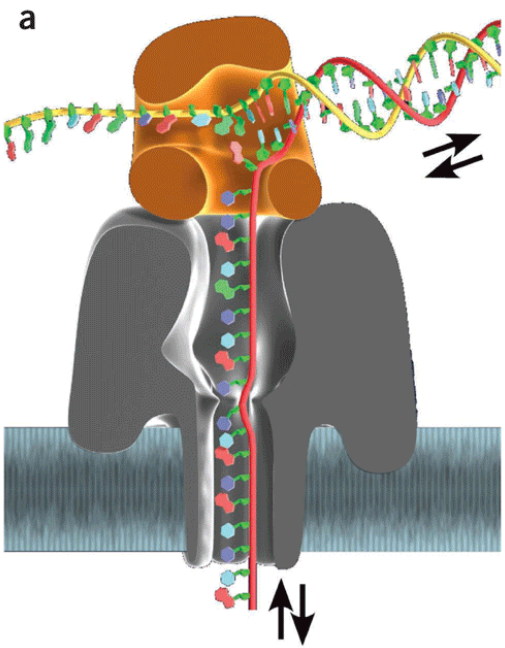
Quang and Xie. DanQ. 2016

RNN application: nanopore base-calling

MinION nanopore sequencer

10-20 Gb of DNA per flow cell
Long reads: 10^3 – 10^5 bps (why does length matter?)
Start kit: $1000

RNN application: nanopore base-calling



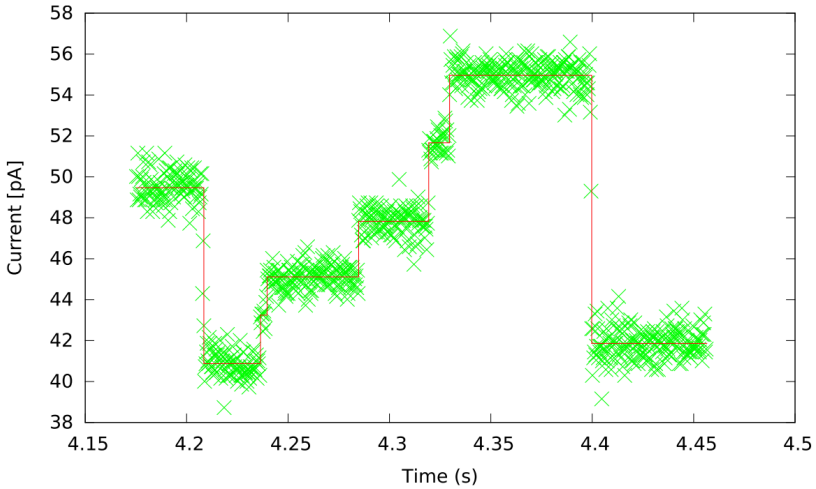


Fig 1. Raw signal from MinION and its segmentation to events. The plot was generated from the *E. coli* data (http://www.ebi.ac.uk/ena/data/view/ERR1147230).

Boza et al. 2017

RNN application: nanopore base-calling

Machine learning challenge: convert the traces of currents into {A, C, T, G} bases.

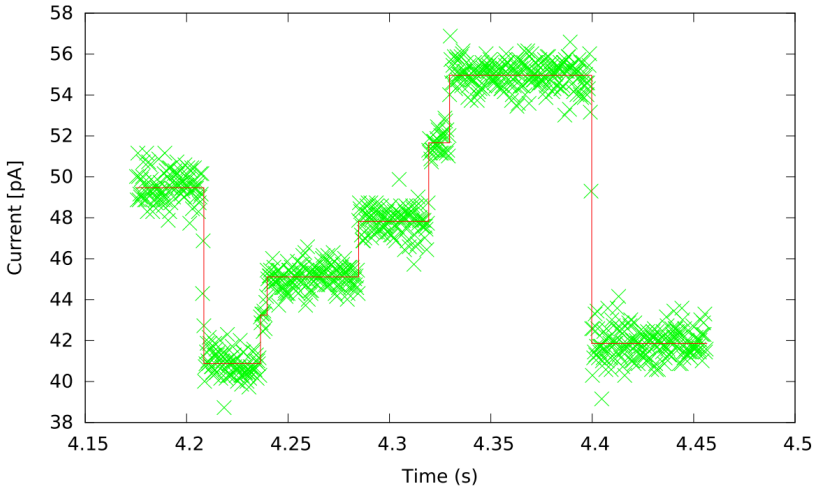
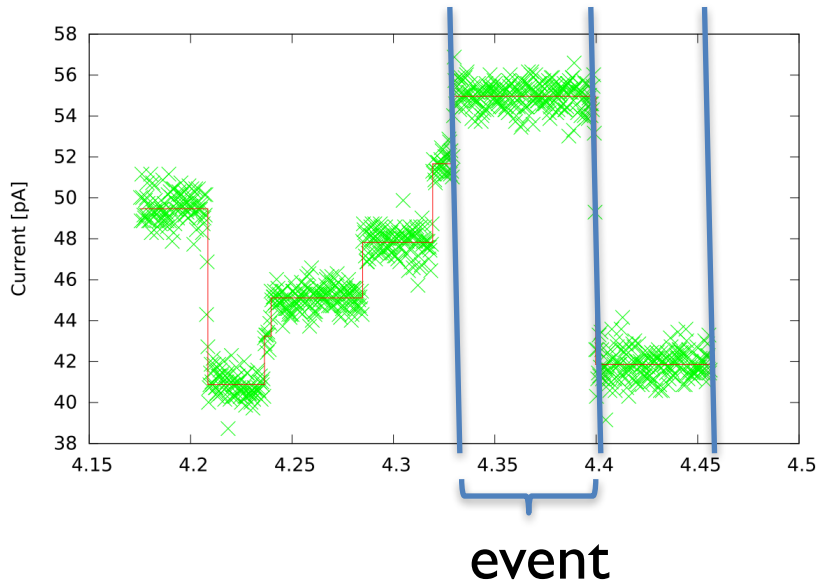


Fig 1. Raw signal from MinION and its segmentation to events. The plot was generated from the *E. coli* data (http://www.ebi.ac.uk/ena/data/view/ERR1147230).

Boza et al. 2017

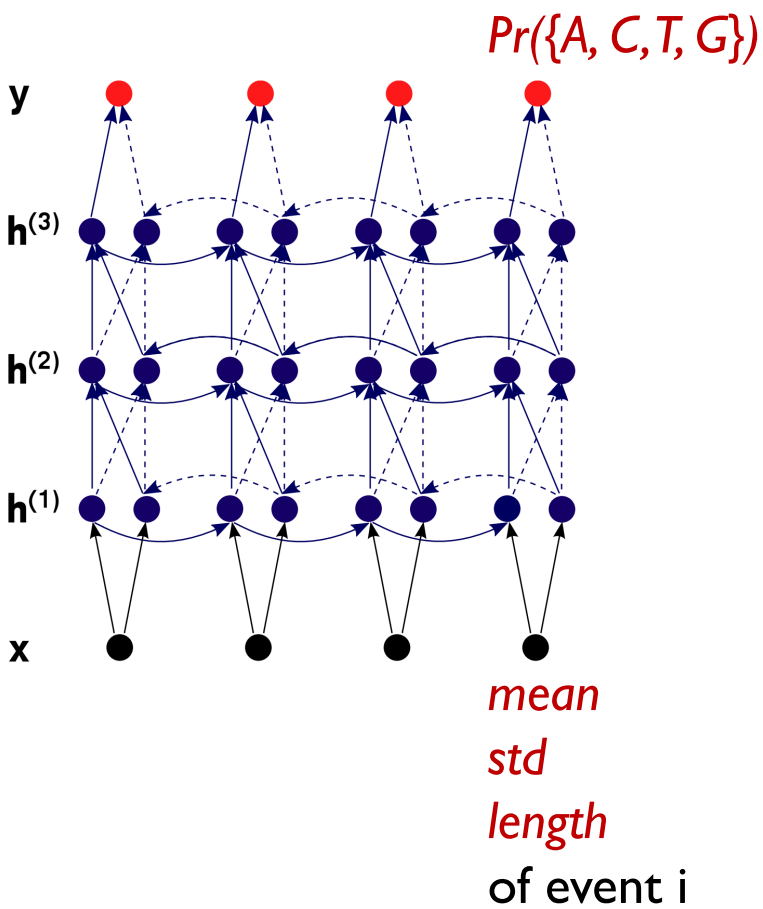
RNN application: nanopore base-calling

Machine learning challenge: convert the traces of currents into {A, C, T, G} bases.



Boza et al. 2017

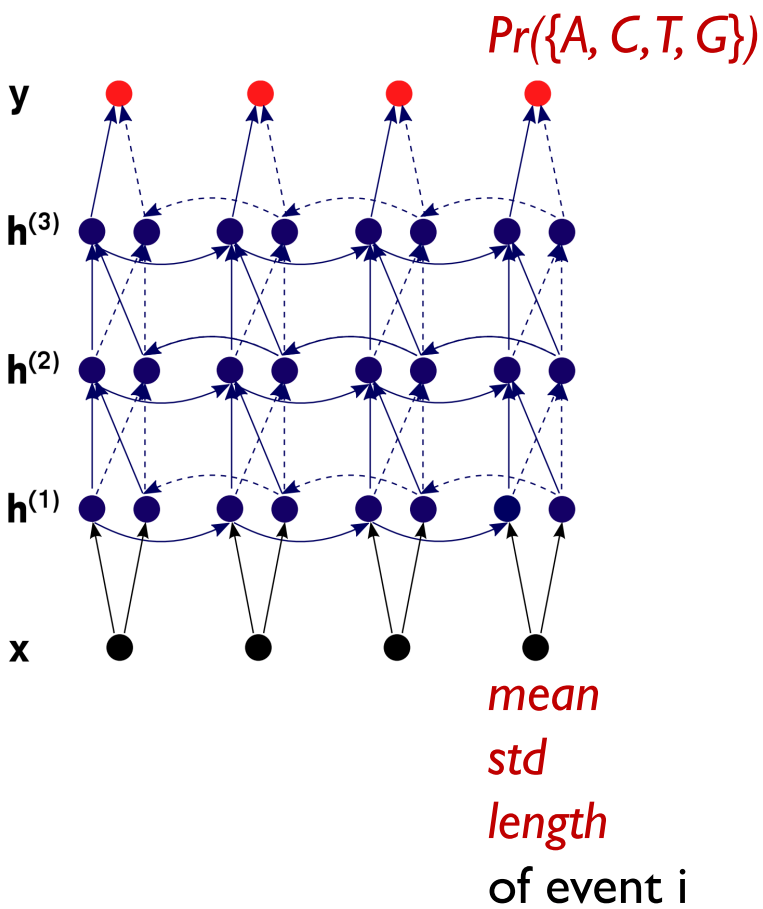
RNN application: nanopore base-calling



$$
\begin{aligned}
\vec{h}_i^{(1)} &= f_1(\vec{h}_{i-1}^{(1)}, \vec{x}_i) \\
\vec{h}_i^{(2)} &= f_2(\vec{h}_{i-1}^{(2)}, \vec{h}_i^{(1)}) \\
\vec{h}_i^{(3)} &= f_3(\vec{h}_{i-1}^{(3)}, \vec{h}_i^{(2)}) \\
\vec{y}_i &= g(\vec{h}_i^{(3)})
\end{aligned}
$$

Boza et al. 2017

RNN application: nanopore base-calling

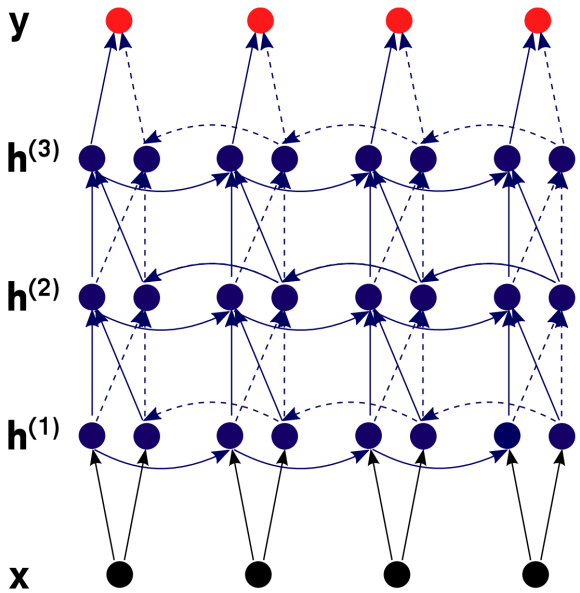


$$
\begin{aligned}
\vec{h}_i^{(1+)} &= f_{1+}(\vec{h}_{i-1}^{(1+)}, \vec{x}_i) \\
\vec{h}_i^{(1-)} &= f_{1-}(\vec{h}_{i+1}^{(1-)}, \vec{x}_i) \\
\vec{h}_i^{(1)} &= \vec{h}_i^{(1+)} || \vec{h}_i^{(1-)} \\
\vec{h}_i^{(2+)} &= f_{2+}(\vec{h}_{i-1}^{(2+)}, \vec{h}_i^{(1)}) \\
\vec{h}_i^{(2-)} &= f_{2-}(\vec{h}_{i+1}^{(2-)}, \vec{h}_i^{(1)}) \\
\vec{h}_i^{(2)} &= \vec{h}_i^{(2+)} || \vec{h}_i^{(2-)} \\
\vec{y}_i &= g(\vec{h}_i^{(2)})
\end{aligned}
$$

Boza et al. 2017

RNN application: nanopore base-calling

Pr({A, C, T, G})



mean
std
length
of event i

Table 1. Accuracy of base callers on two R7.3 testing data sets. The results of base calling were aligned to the reference using BWA-MEM [28]. The accuracy was computed as the number of matches in the alignment divided by the length of the alignment.

	E. coli	*K. pneumoniae*
Template reads		
Metrichor	71.3%	68.1%
Nanocall	68.3%	67.5%
DeepNano	77.9%	76.3%

Boza et al. 2017