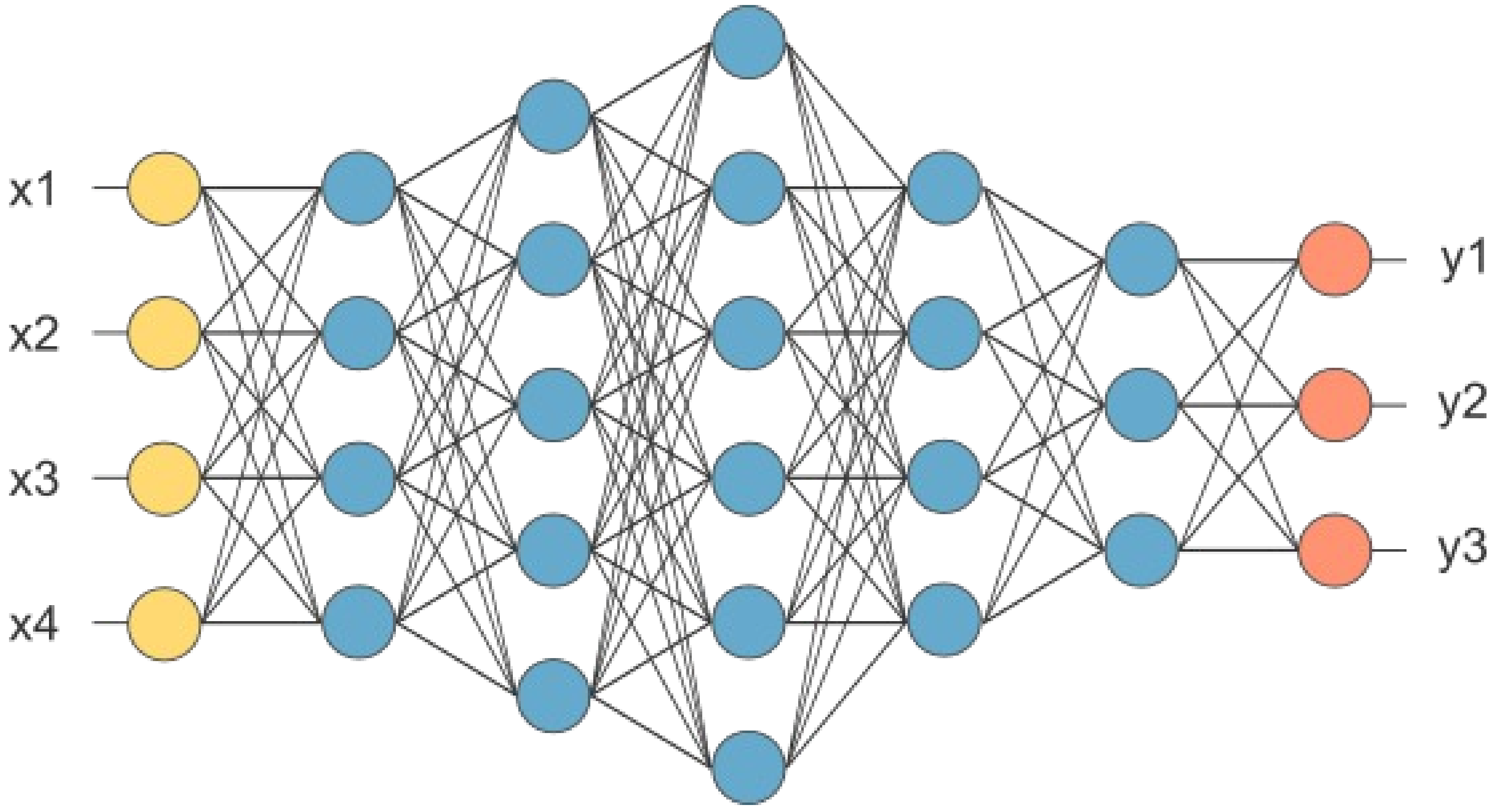
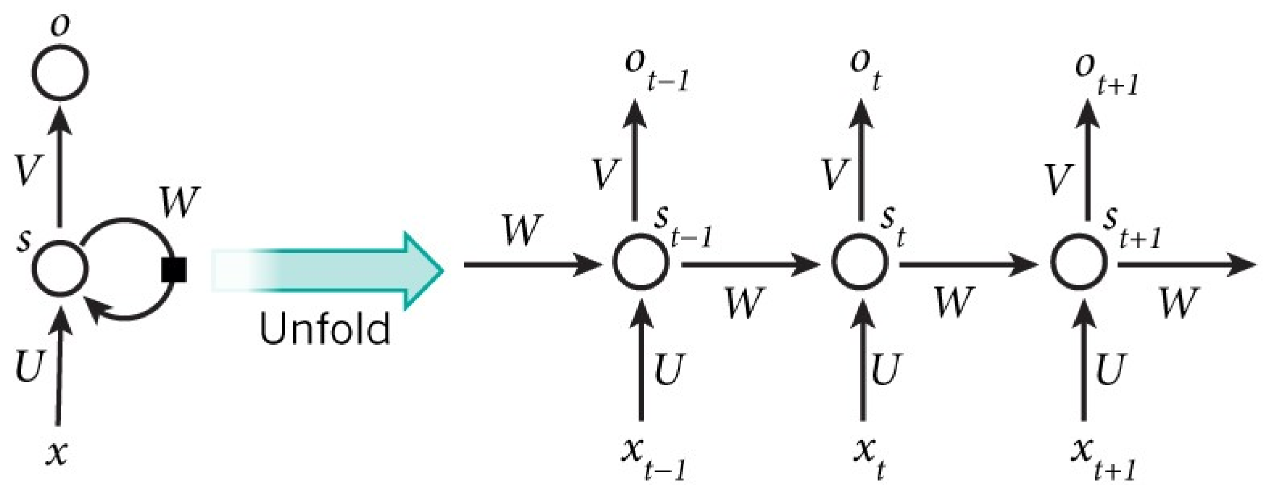
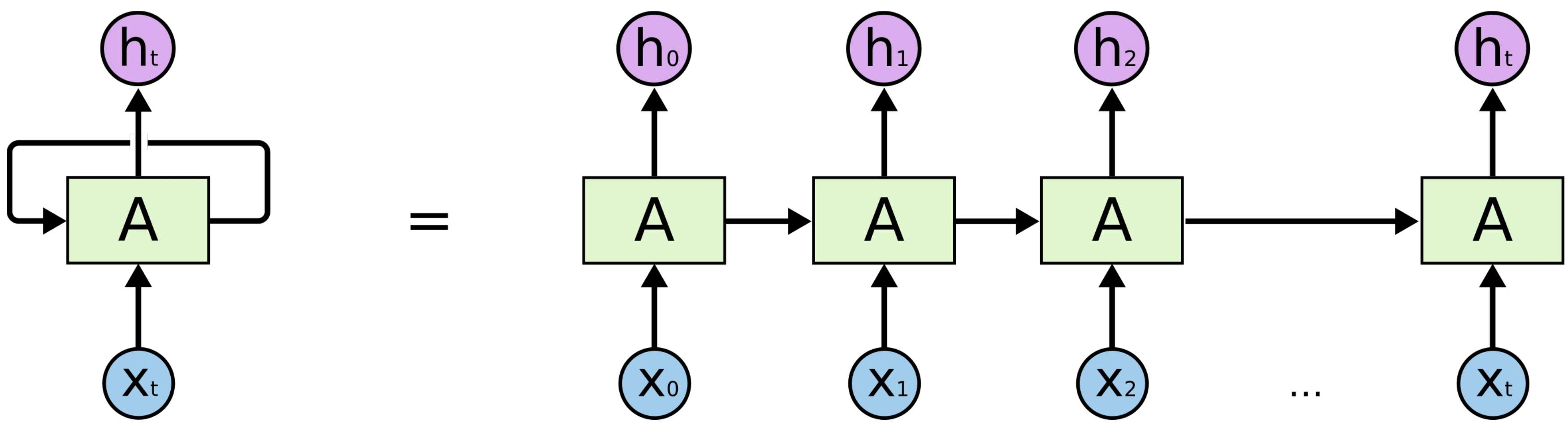
Lecture 6 Smaller Network: RNN



This is our fully connected network. If x_1 ... x_n, n is very large and growing, this network would become too large. We now will input one x_i at a time, and re-use the same edge weights.

Recurrent Neural Network

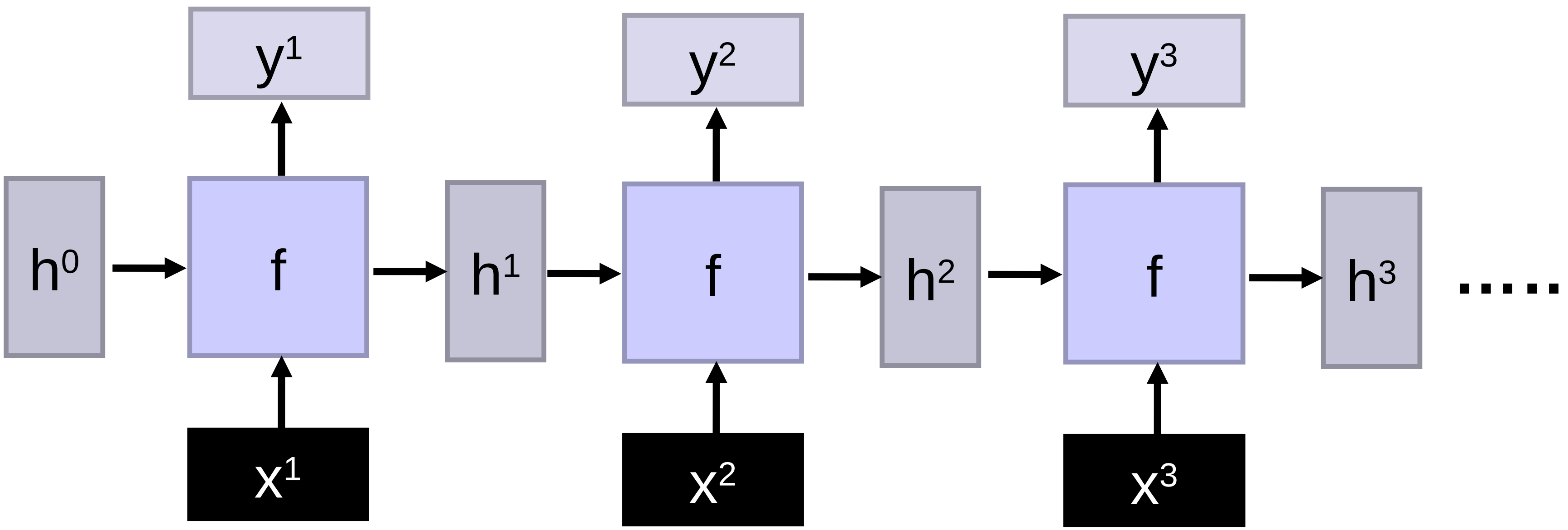




How does RNN reduce complexity?

- Given function f: h',y=f(h,x)

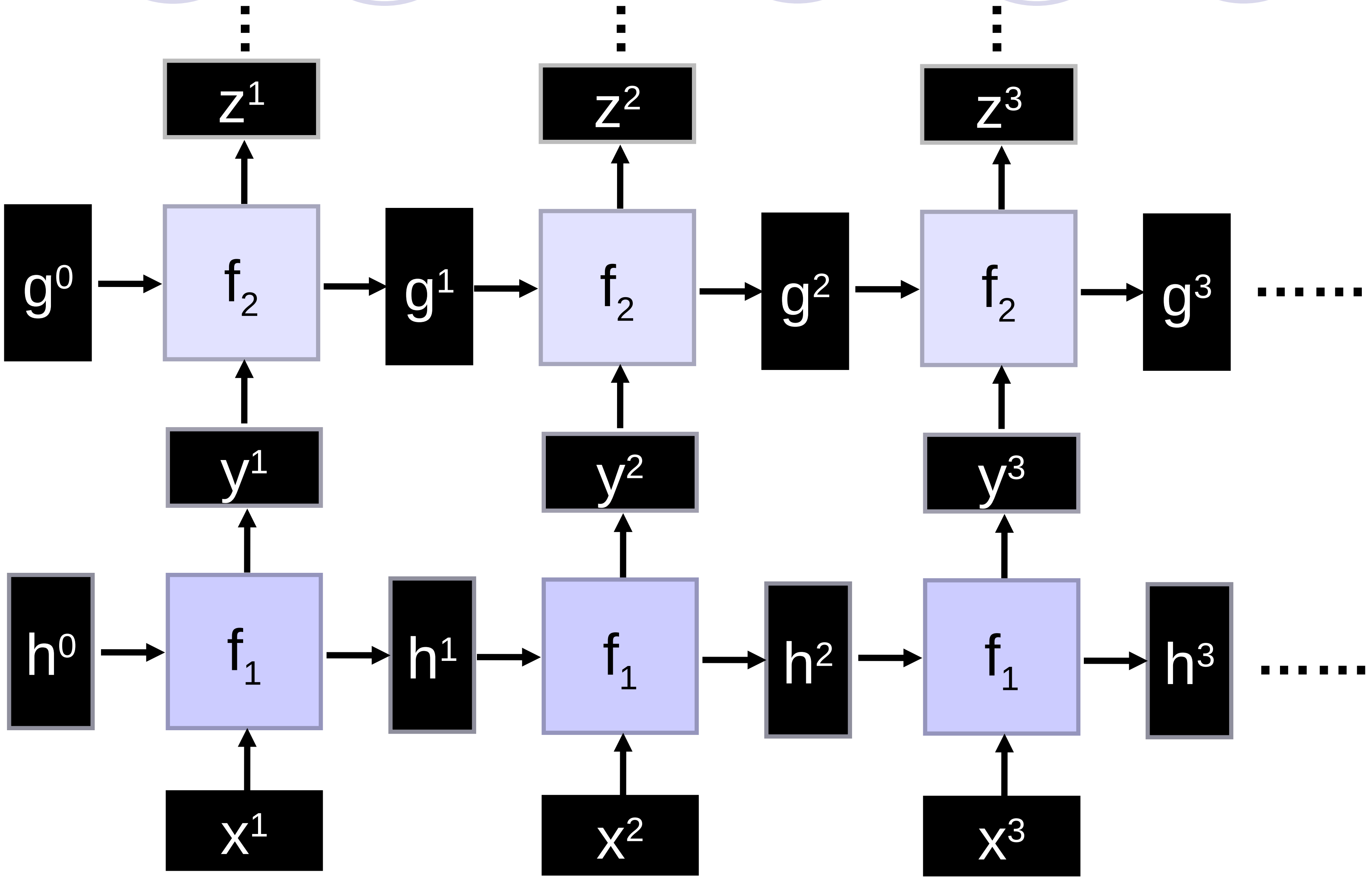
h and h' are vectors with the same dimension



No matter how long the input/output sequence is, we only need one function f. If f's are different, then it becomes a feedforward NN. This may be treated as another compression from fully connected network.

Deep RNN

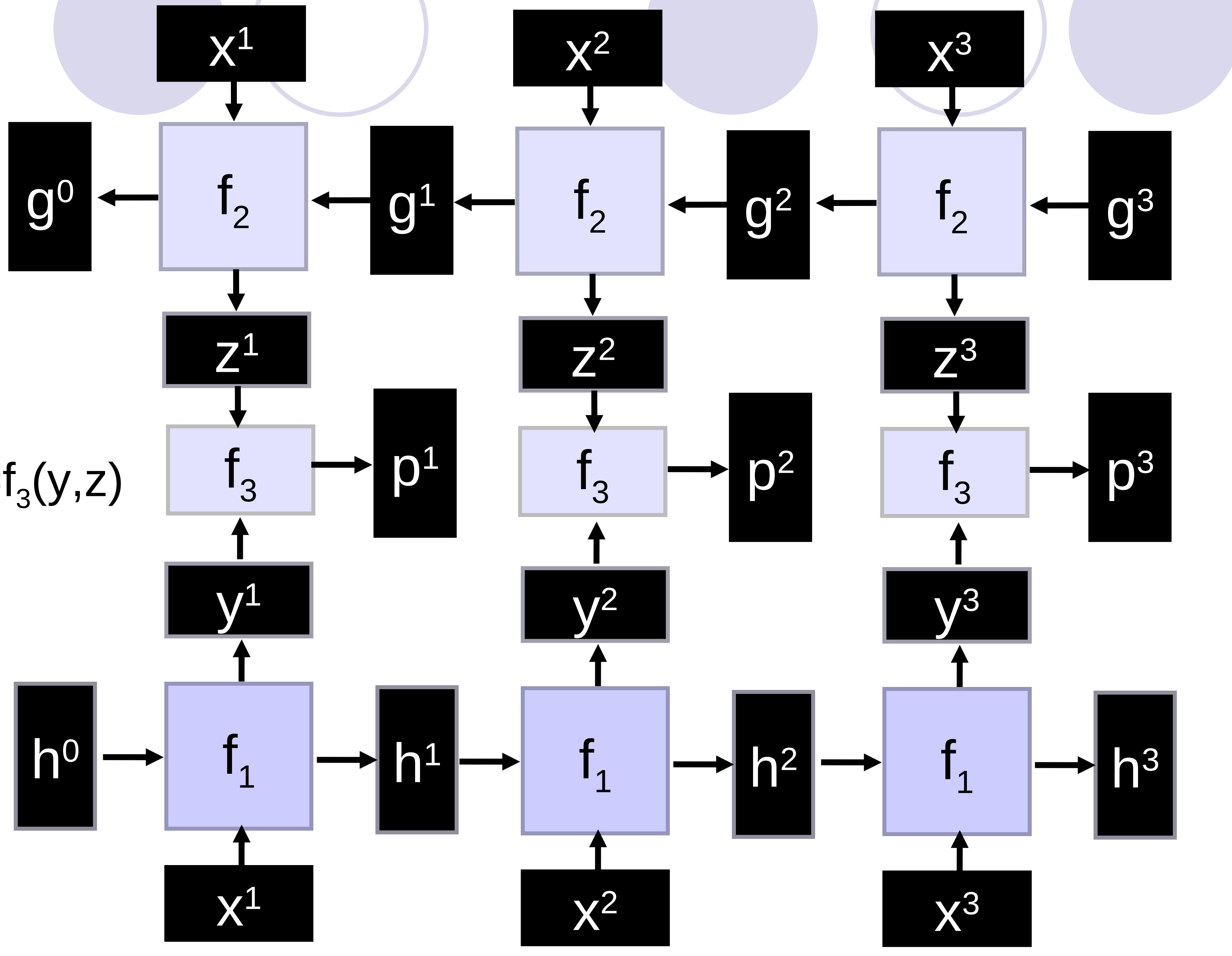
$h',y = f_1(h,x)$, $g',z = f_2(g,y)$ …



Bidirectional RNN

$y,h=f_1(x,h)$ $z,g = f_2(g,x)$

$p=f_3(y,z)$

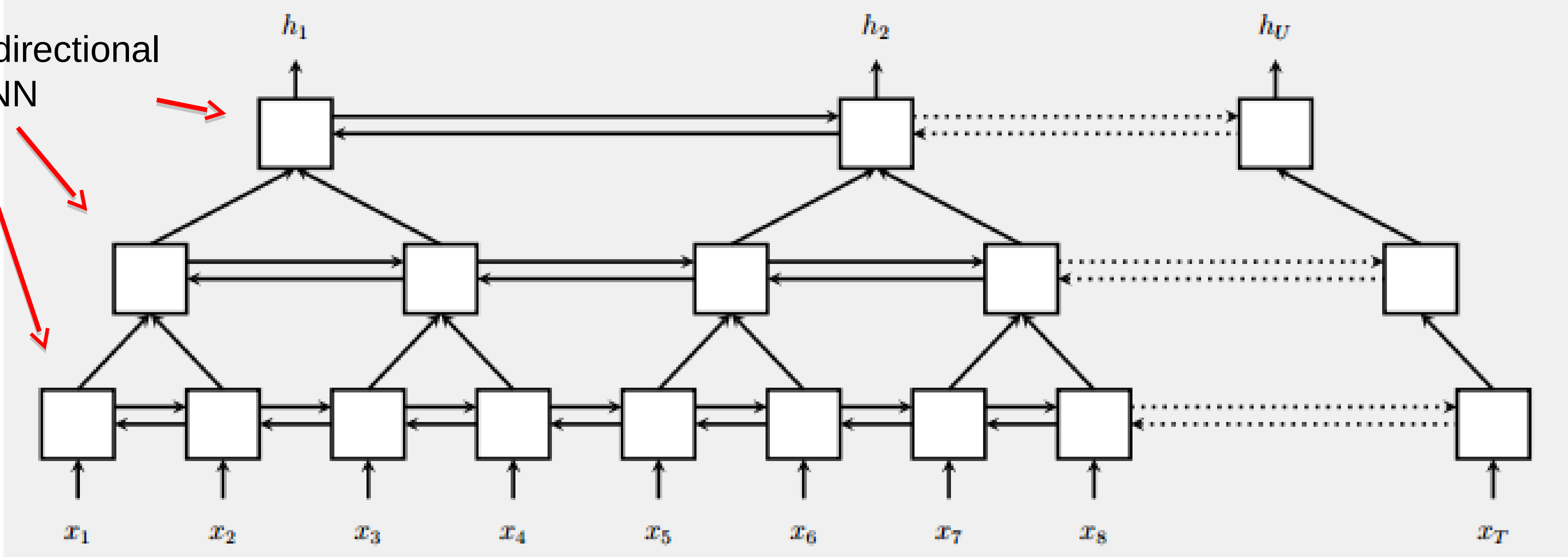


Pyramid RNN

Significantly speed up training

- Reducing the number of time steps

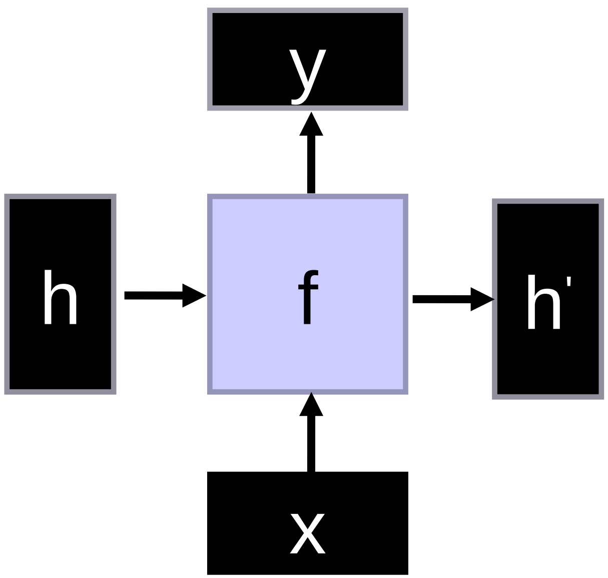
Bidirectional RNN



W. Chan, N. Jaitly, Q. Le and O. Vinyals, “Listen, attend and spell: A neural network for large vocabulary conversational speech recognition,” ICASSP, 2016

Naïve RNN

- Given function f: h',y=f(h,x)



$$h' = \sigma(W^h h + W^i x)$$

$$y = \sigma(W^o h')$$

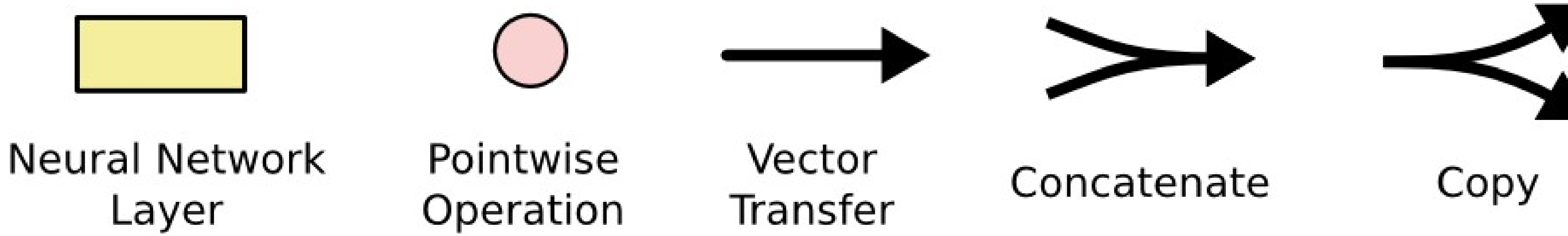
softmax

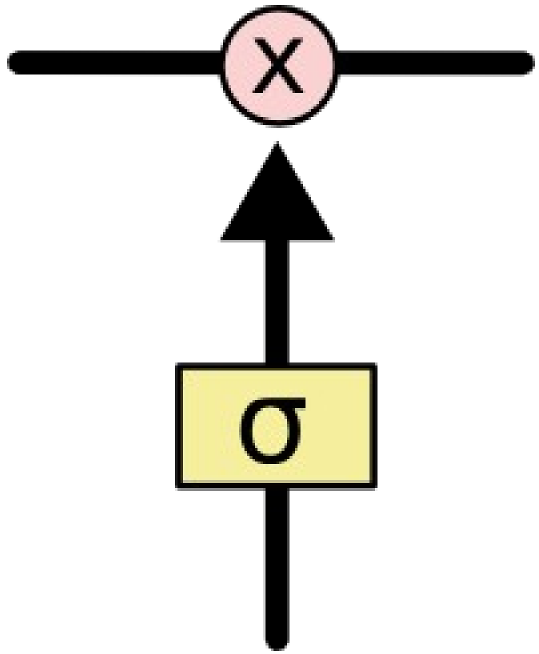
Note, y is computed from h'

We have ignored the bias

Problems with naive RNN

- When dealing with a time series, it tends to forget old information. When there is a distant relationship of unknown length, we wish to have a “memory” to it.
- Vanishing gradient problem.





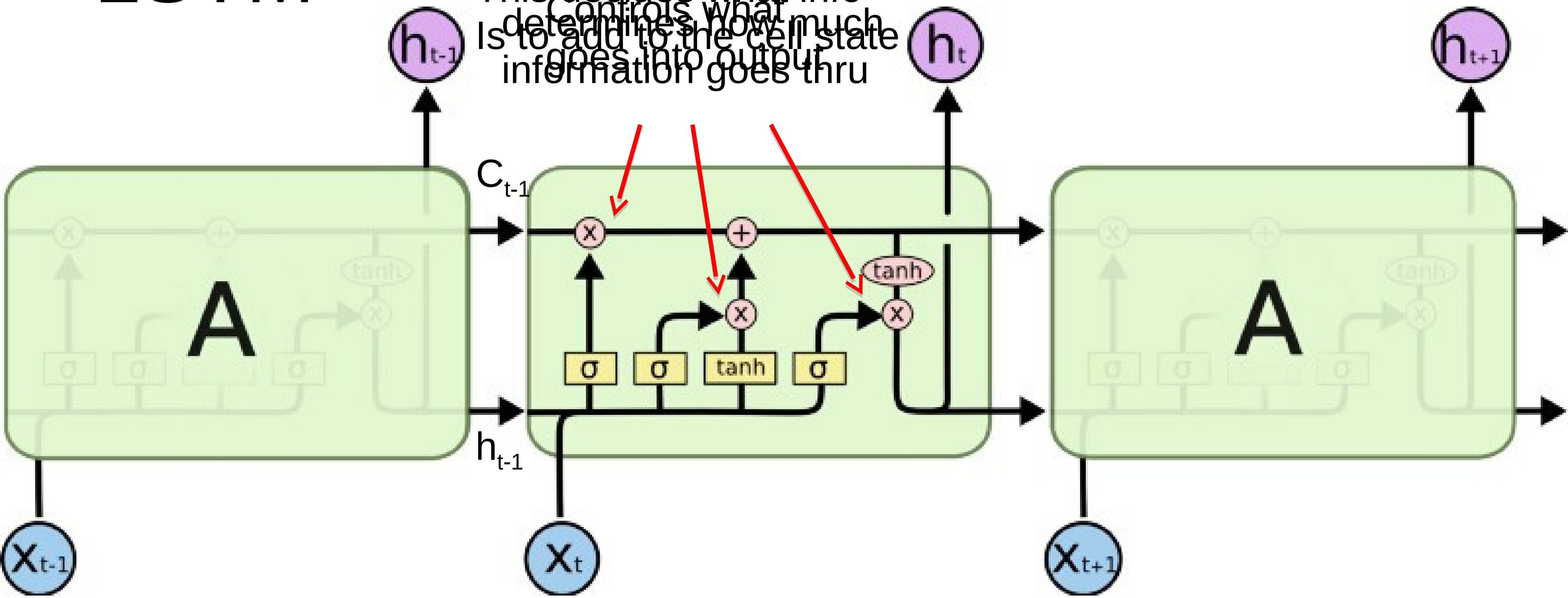
The sigmoid layer outputs numbers between 0-1 determine how much each component should be let through. Pink X gate is point-wise multiplication.

LSTM

Output gate
Controls what
goes into output

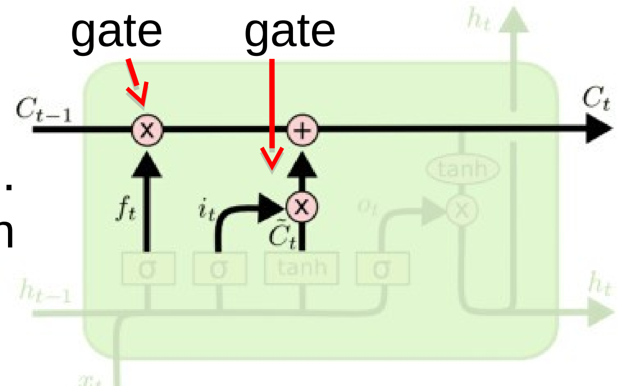
This sigmoid gate
determines how much
information goes thru

This decides what info
Is to add to the cell state



Forget gate

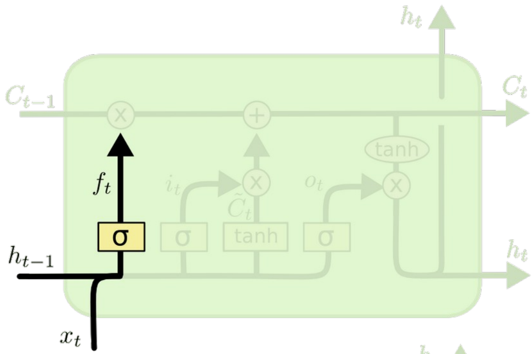
input gate



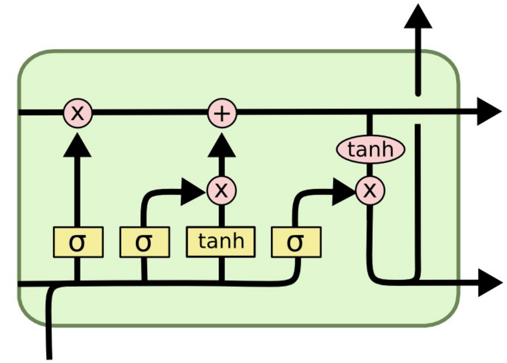
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

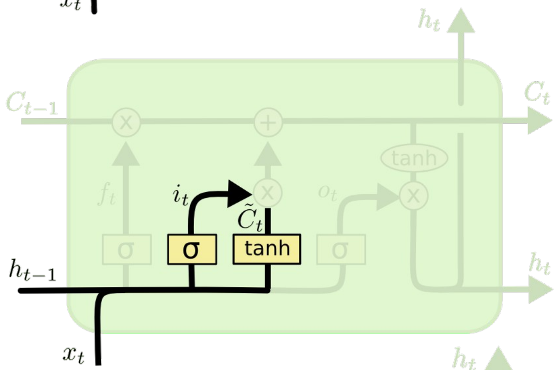
The core idea is this cell state C_t, it is changed slowly, with only minor linear interactions. It is very easy for information to flow along it unchanged.

Why sigmoid or tanh:
Sigmoid: 0,1 gating as switch.
Vanishing gradient problem in LSTM is handled already.
ReLU replaces tanh ok?



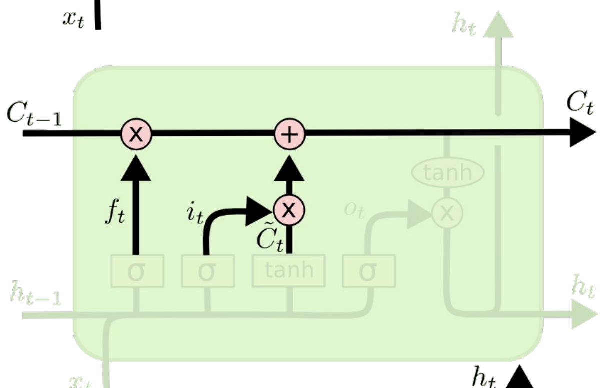
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$





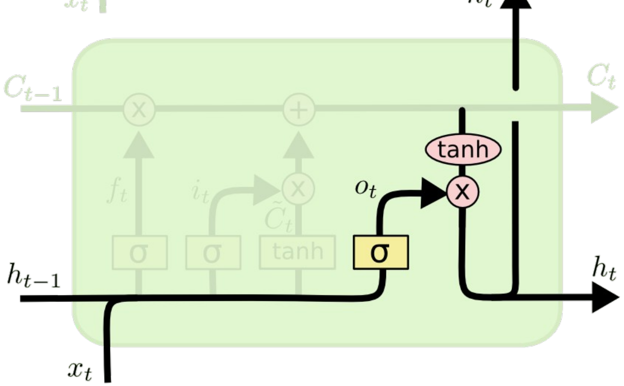
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

i_t decides what component is to be updated.
C'_t provides change contents



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

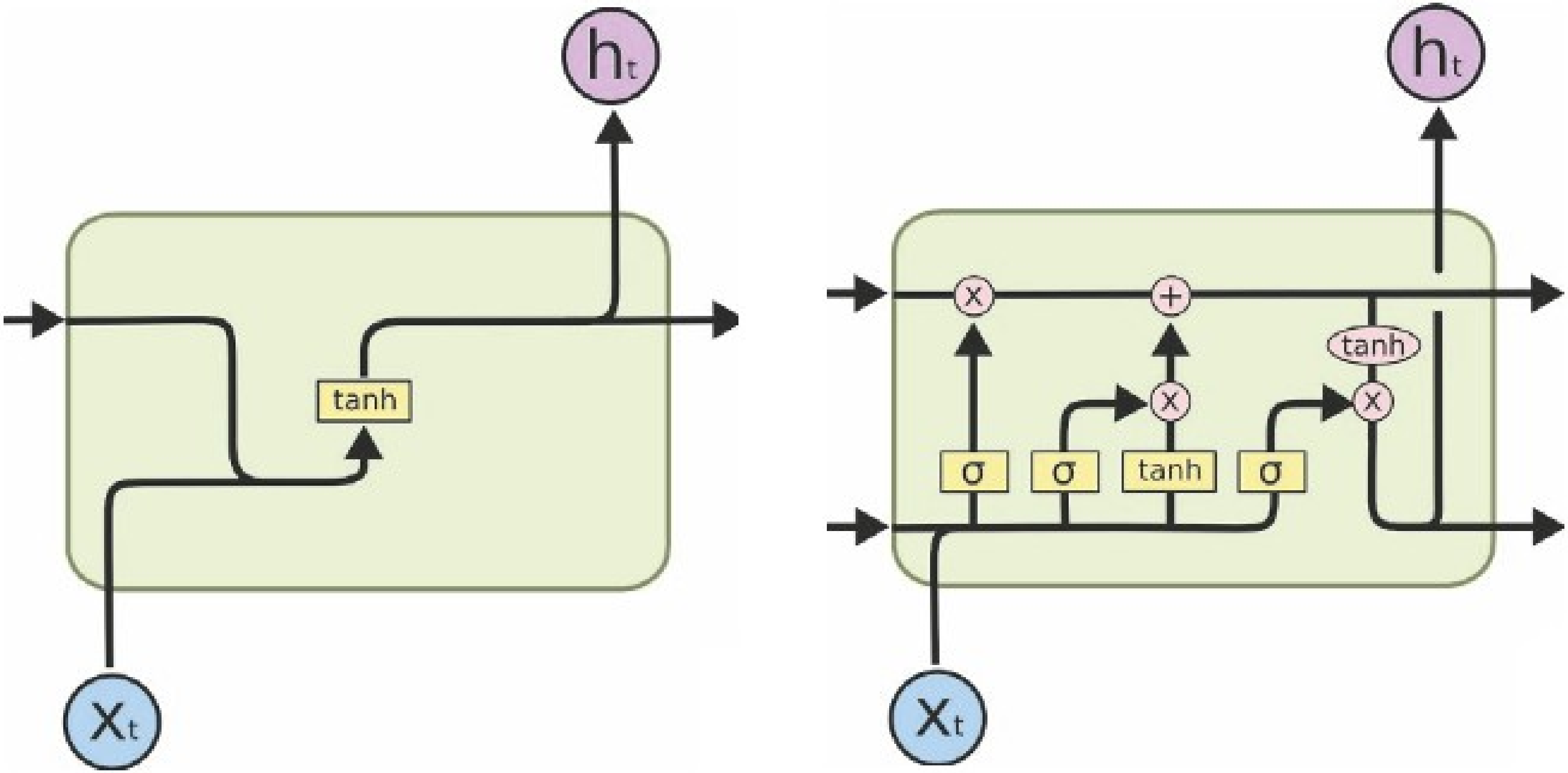
Updating the cell state



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

Decide what part of the cell state to output

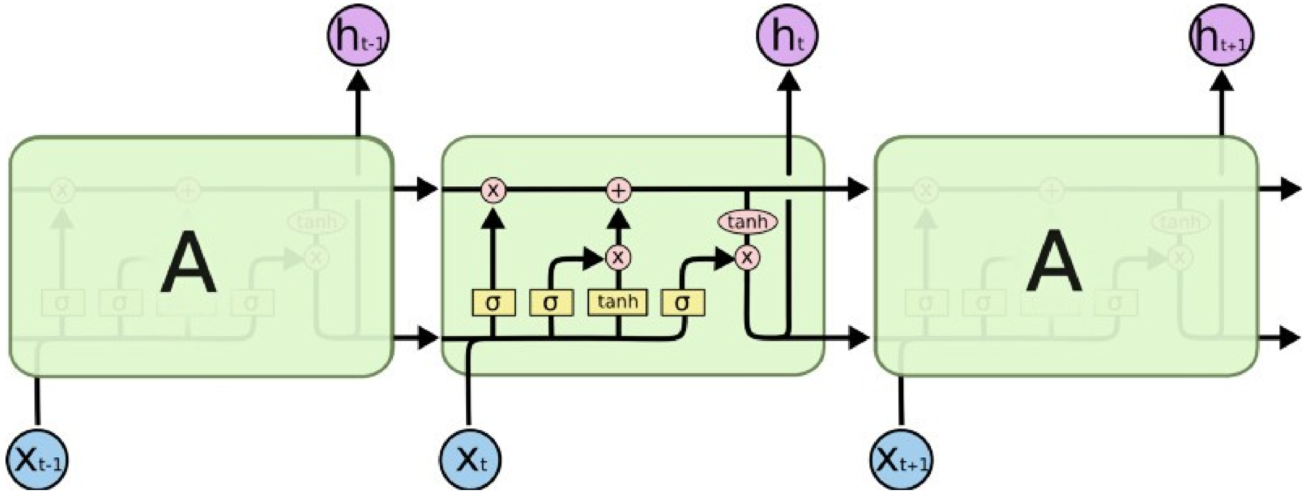
RNN vs LSTM



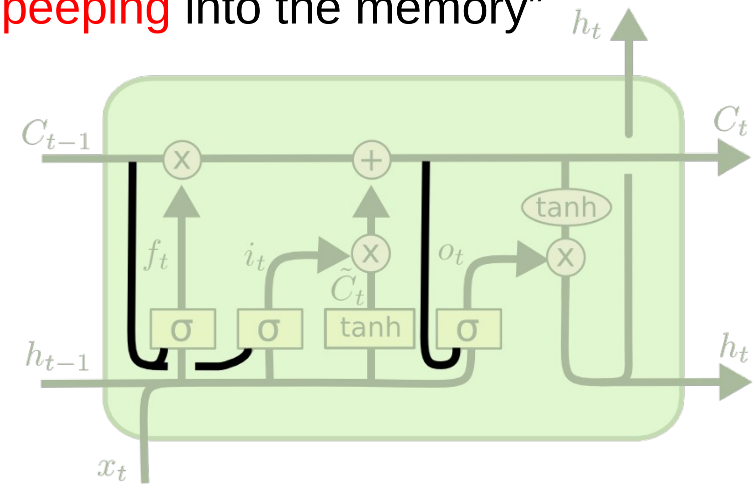
(a) RNN

(b) LSTM

Peephole LSTM



Allows “peeping into the memory”

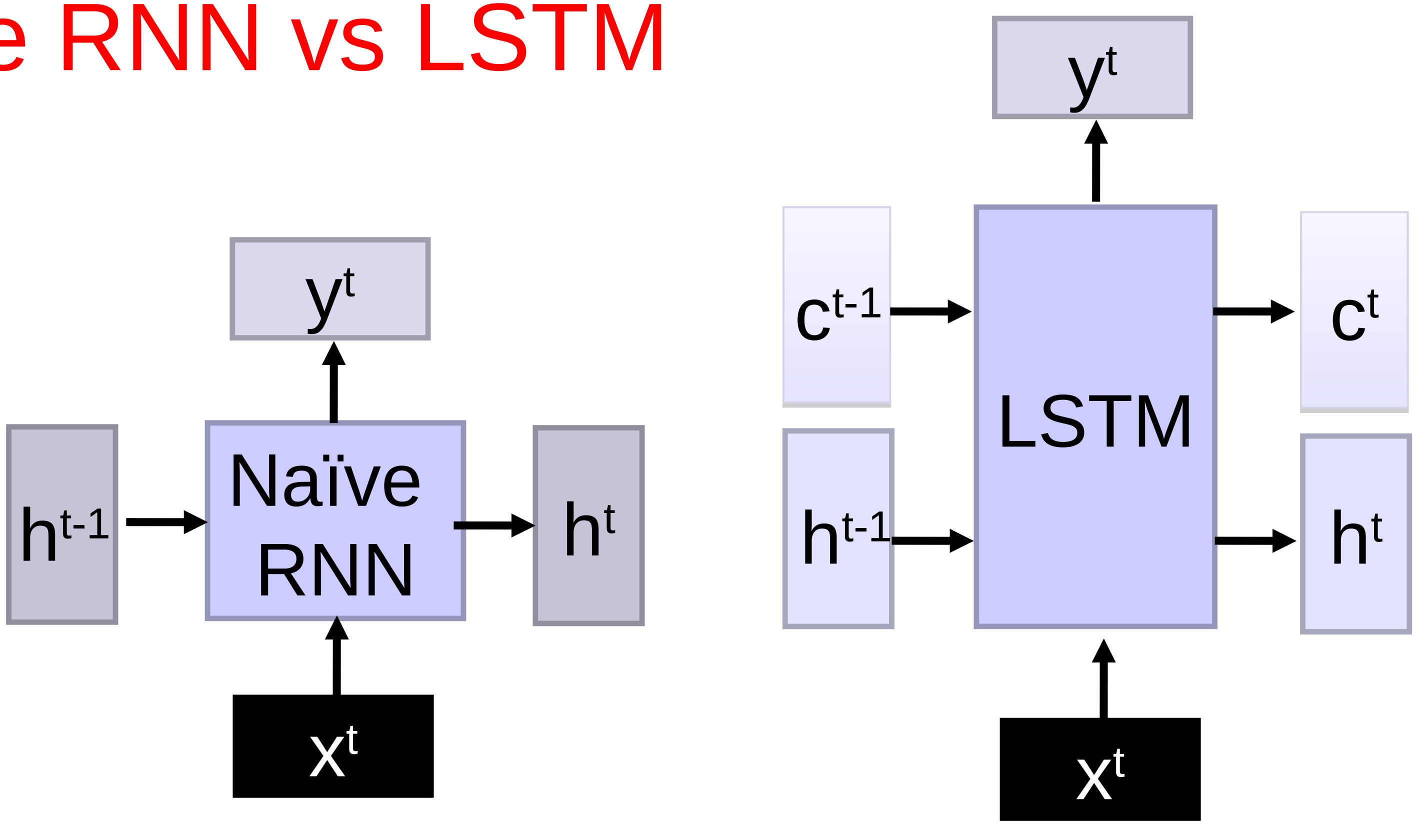


$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_f\right)$$

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

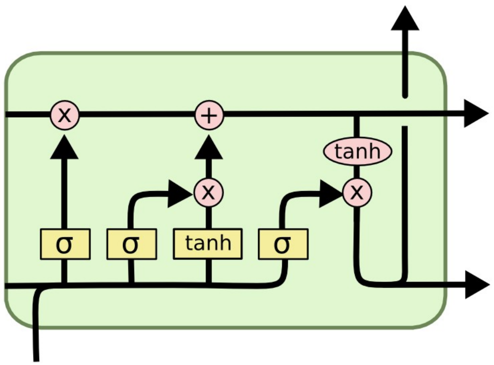
$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] \; + \; b_o\right)$$

Naïve RNN vs LSTM



c changes slowly ➡ c^t is c^{t-1} added by something

h changes faster ➡ h^t and h^{t-1} can be very different



These 4 matrix computation should be done concurrently.

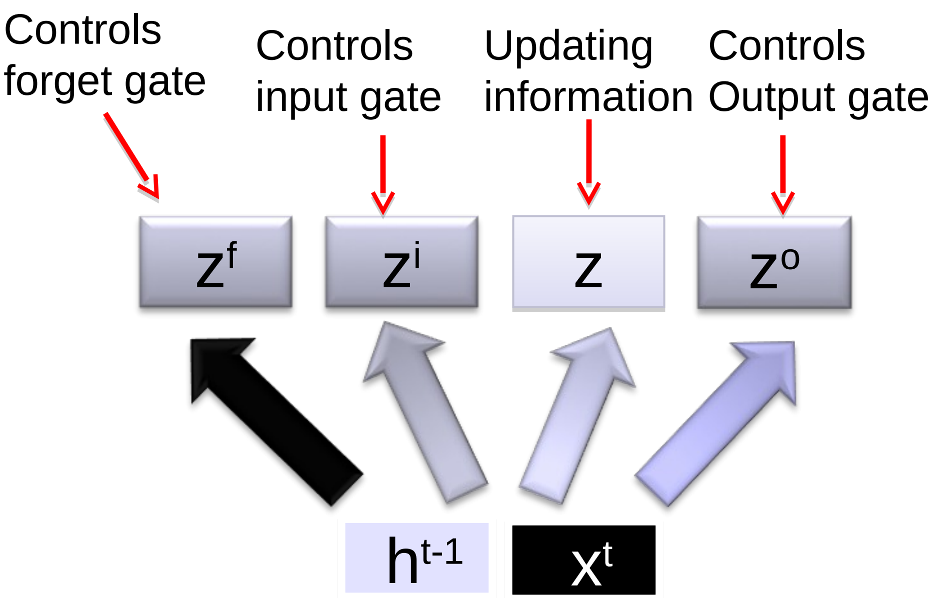
c^{t-1}

$$z = \tanh\left(W \begin{bmatrix} x^t \\ h^{t-1} \end{bmatrix} \right)$$

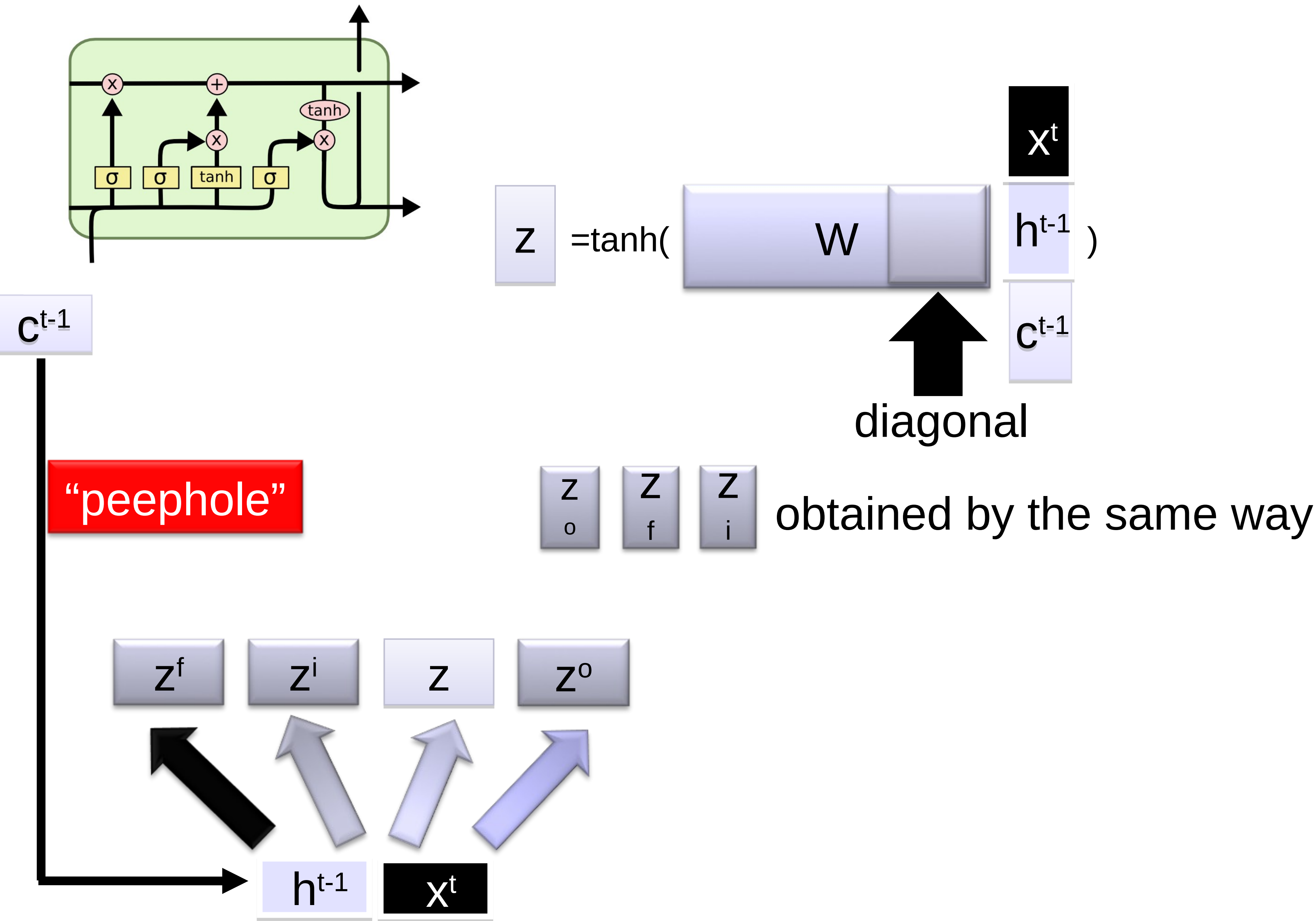
$$z^i = \sigma\left(W^i \begin{bmatrix} x^t \\ h^{t-1} \end{bmatrix} \right)$$

$$z^f = \sigma\left(W^f \begin{bmatrix} x^t \\ h^{t-1} \end{bmatrix} \right)$$

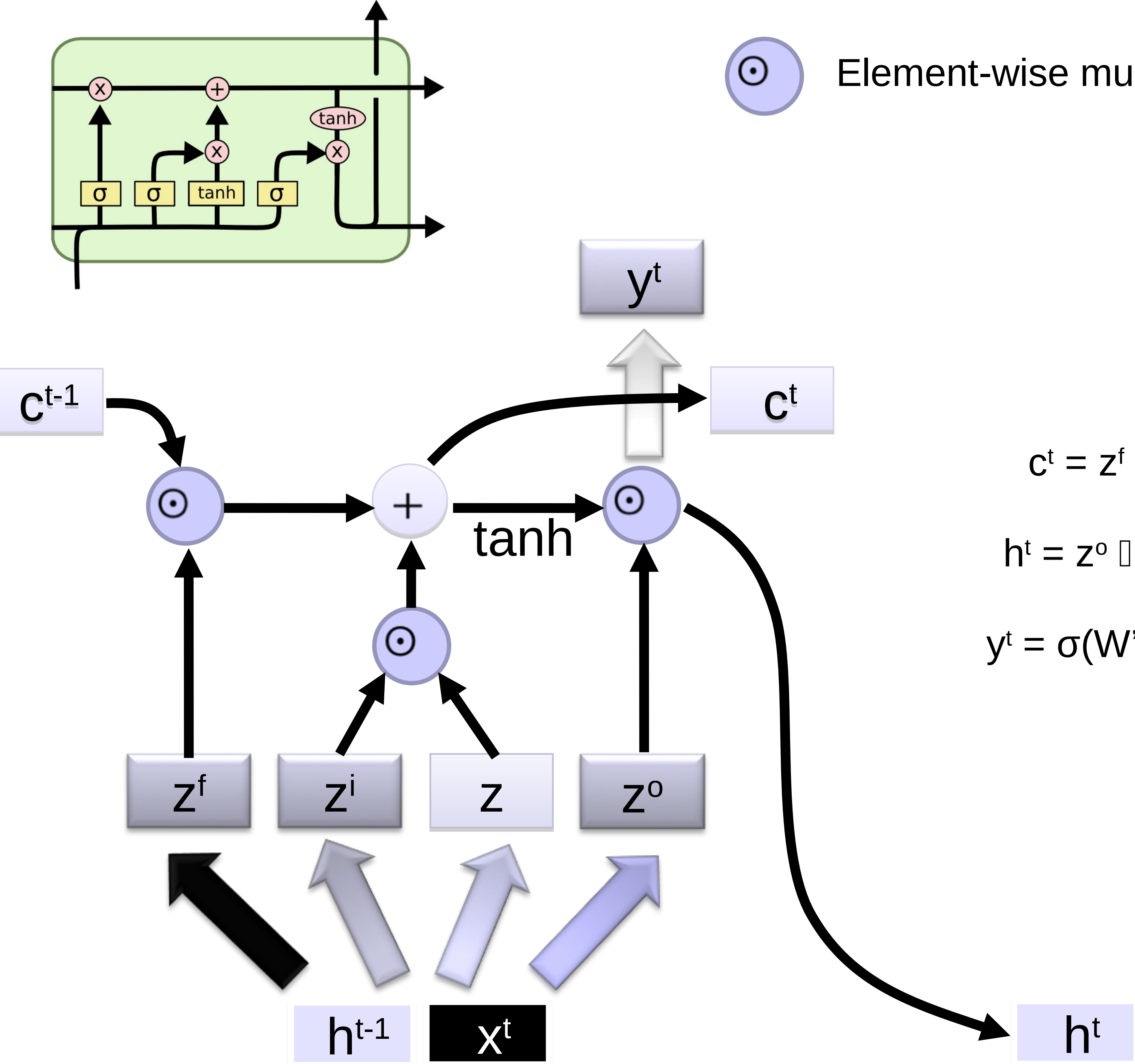
$$z^o = \sigma\left(W^o \begin{bmatrix} x^t \\ h^{t-1} \end{bmatrix} \right)$$



Information flow of LSTM



Information flow of LSTM



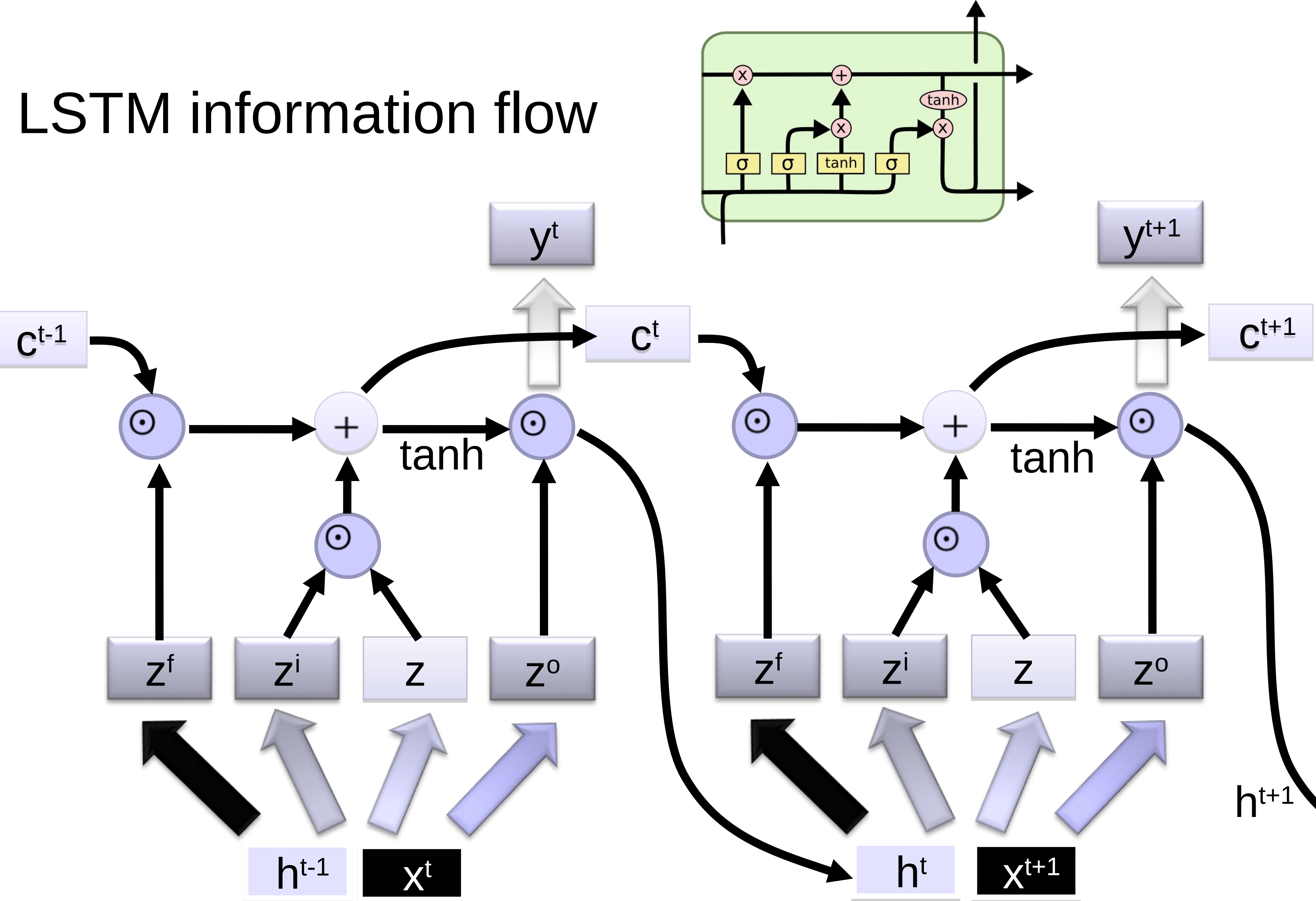
$$c^t = z^f \odot c^{t-1} + z^i \odot z$$

$$h^t = z^o \odot \tanh(c^t)$$

$$y^t = \sigma(W' h^t)$$

Information flow of LSTM

LSTM information flow

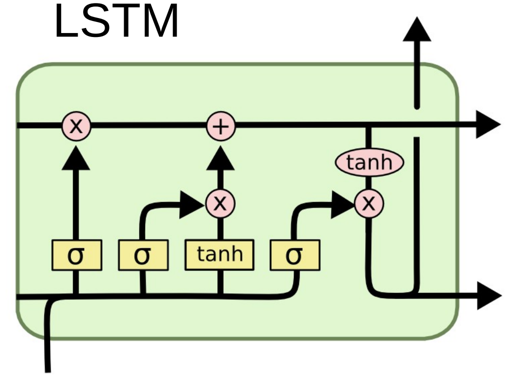


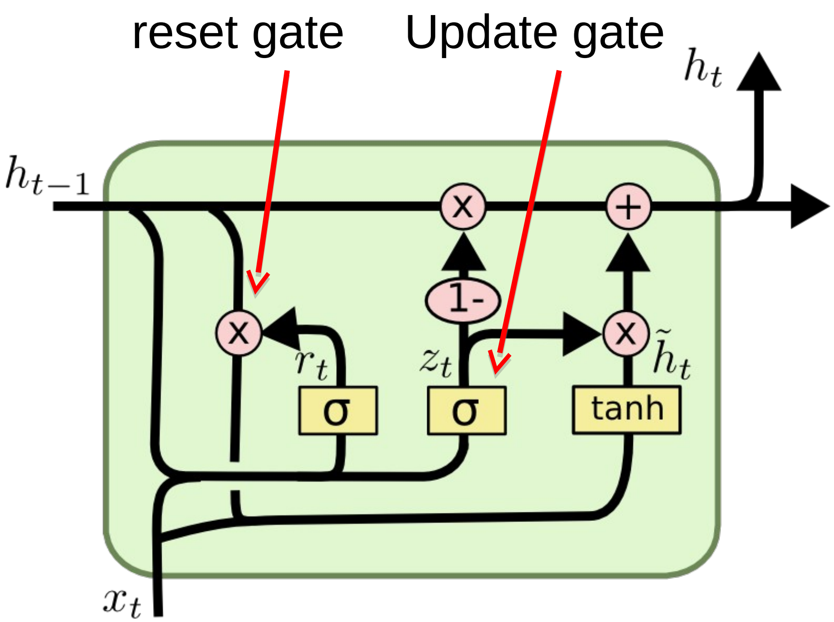
Information flow of LSTM

GRU – gated recurrent unit

(more compression)

LSTM





$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

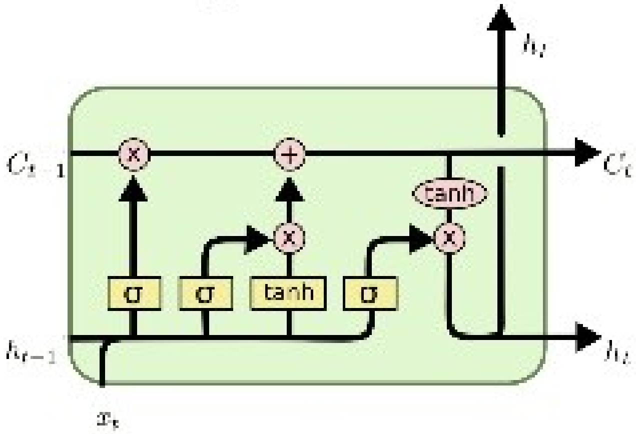
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

It combines the forget and input into a single update gate.
It also merges the cell state and hidden state. This is simpler than LSTM. There are many other variants too.

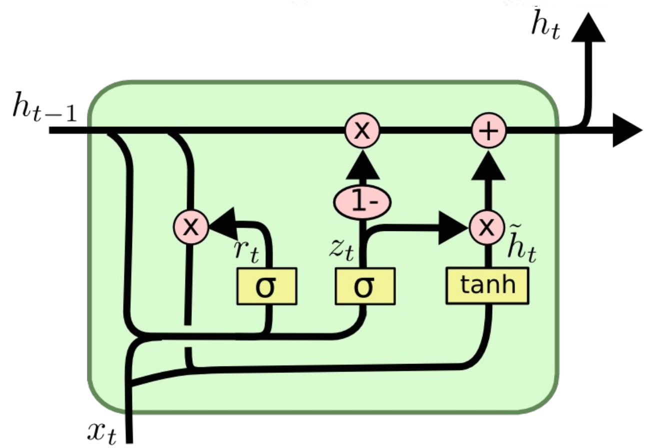
X,*: element-wise multiply

LSTM and GRU

- LSTM [Hochreiter&Schmidhuber97]



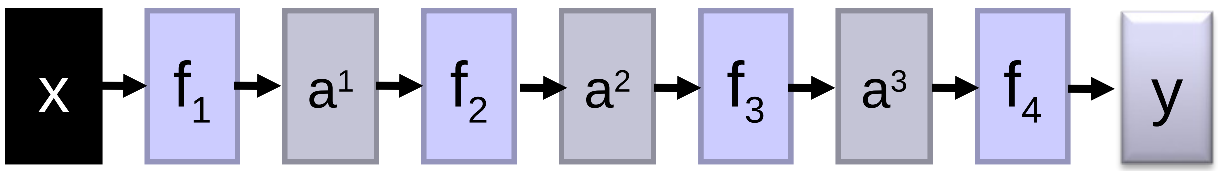
- GRU [Cho+14]



GRUs also takes x_t and h_{t-1} as inputs. They perform some calculations and then pass along h_t. What makes them different from LSTMs is that GRUs don't need the cell layer to pass values along. The calculations within each iteration insure that the h_t values being passed along either retain a high amount of old information or are jump-started with a high amount of new information.

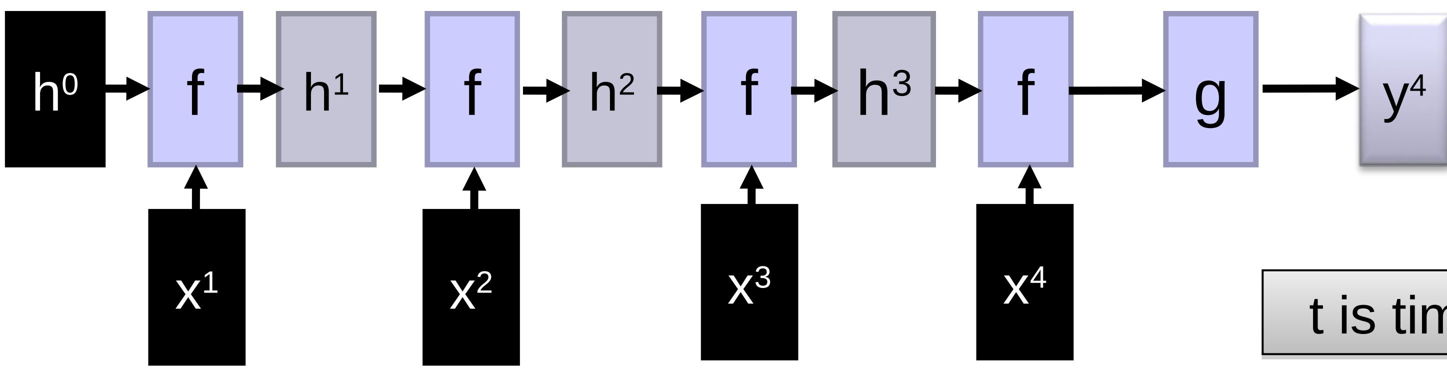
Feed-forward vs Recurrent Network

1. Feedforward network does not have input at each step
2. Feedforward network has different parameters for each layer



t is layer

$$a^t = f_t(a^{t-1}) = \sigma(W^t a^{t-1} + b^t)$$



t is time step

$$a^t = f(a^{t-1}, x^t) = \sigma(W^h a^{t-1} + W^i x^t + b^i)$$

We will turn the recurrent network 90 degrees.

GRU → Highway Network

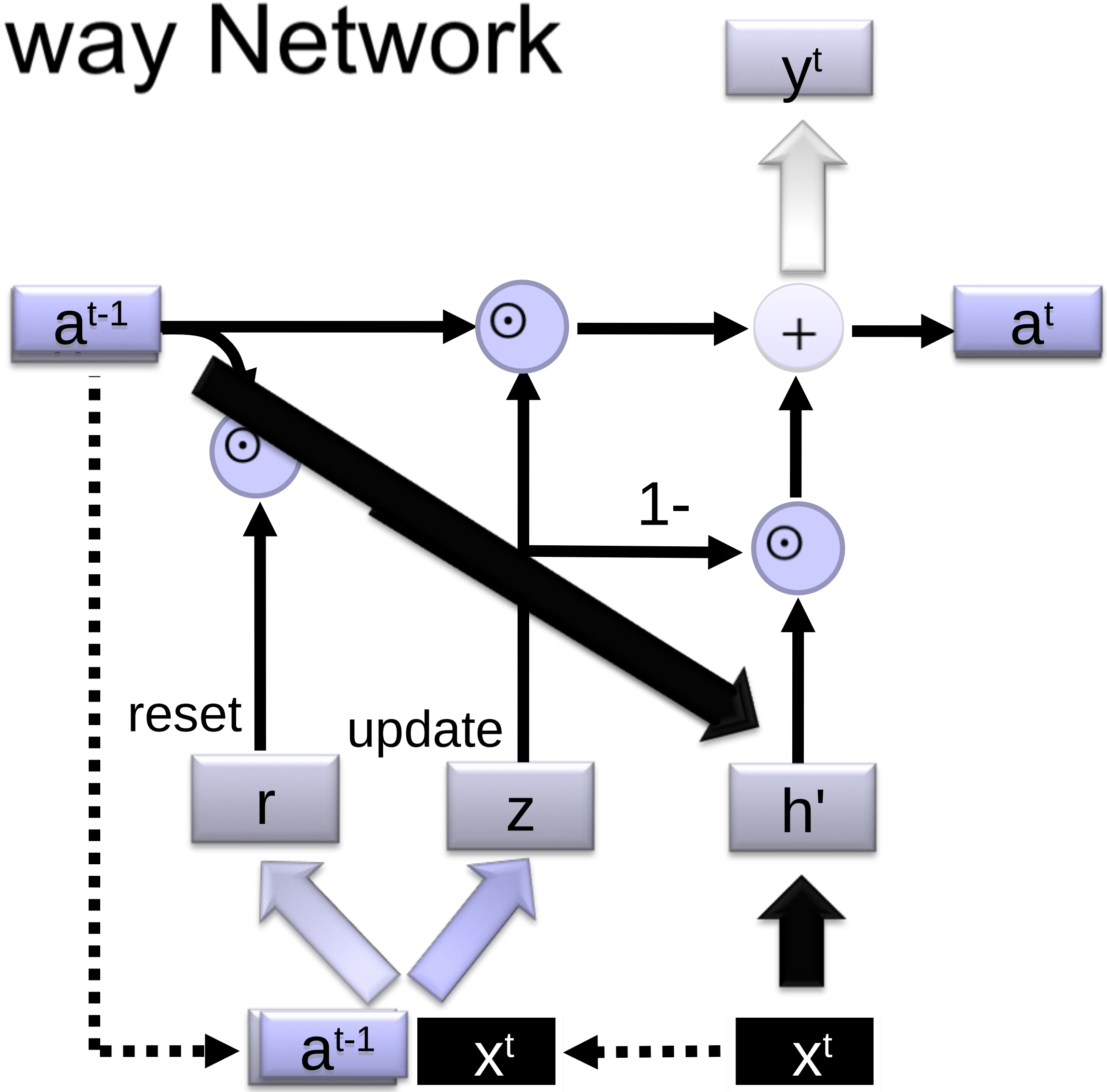
No input x^t at each step

No output y^t at each step

a^{t-1} is the output of the (t-1)-th layer

a^t is the output of the t-th layer

No reset gate



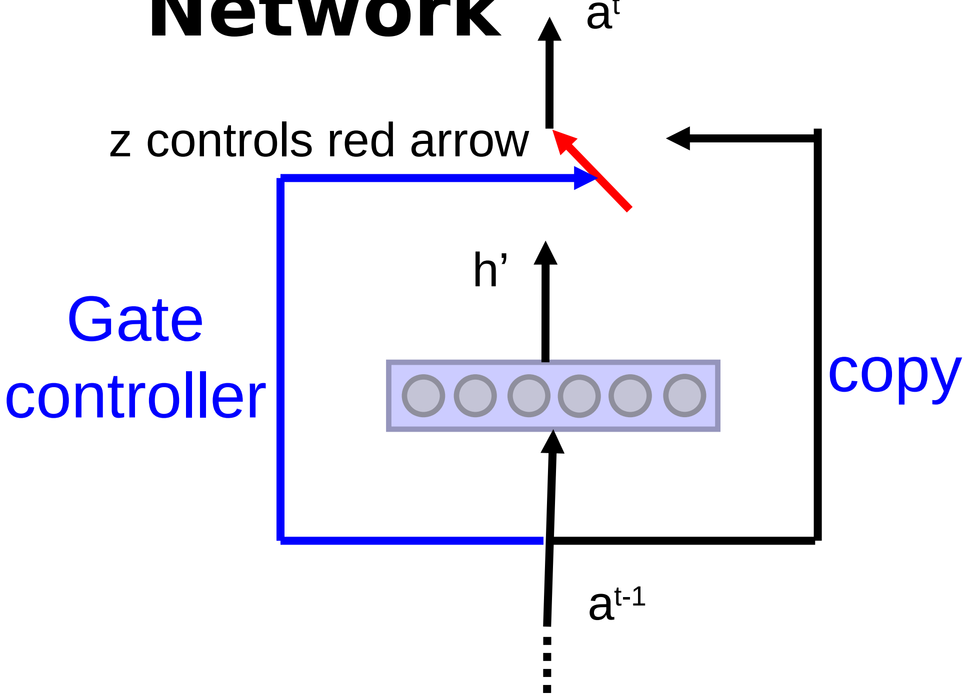
Highway Network

$h' = \sigma(Wa^{t-1})$

$z = \sigma(W'a^{t-1})$

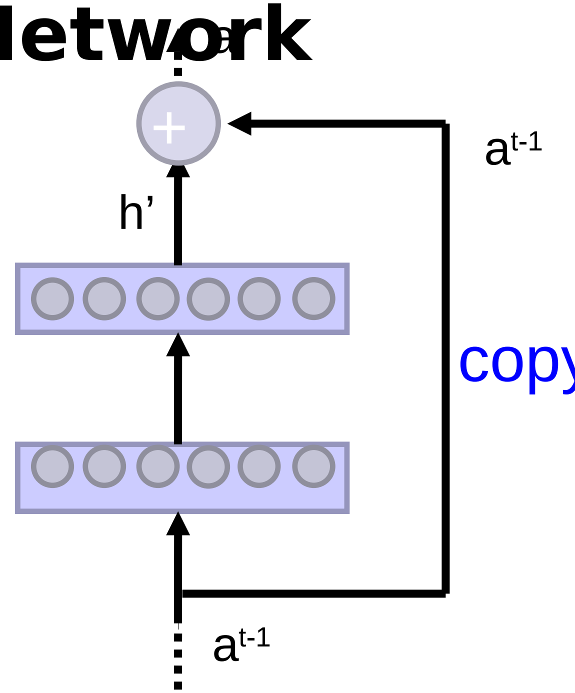
$a^t = z \odot a^{t-1} + (1-z) \odot h$

- **Highway Network**

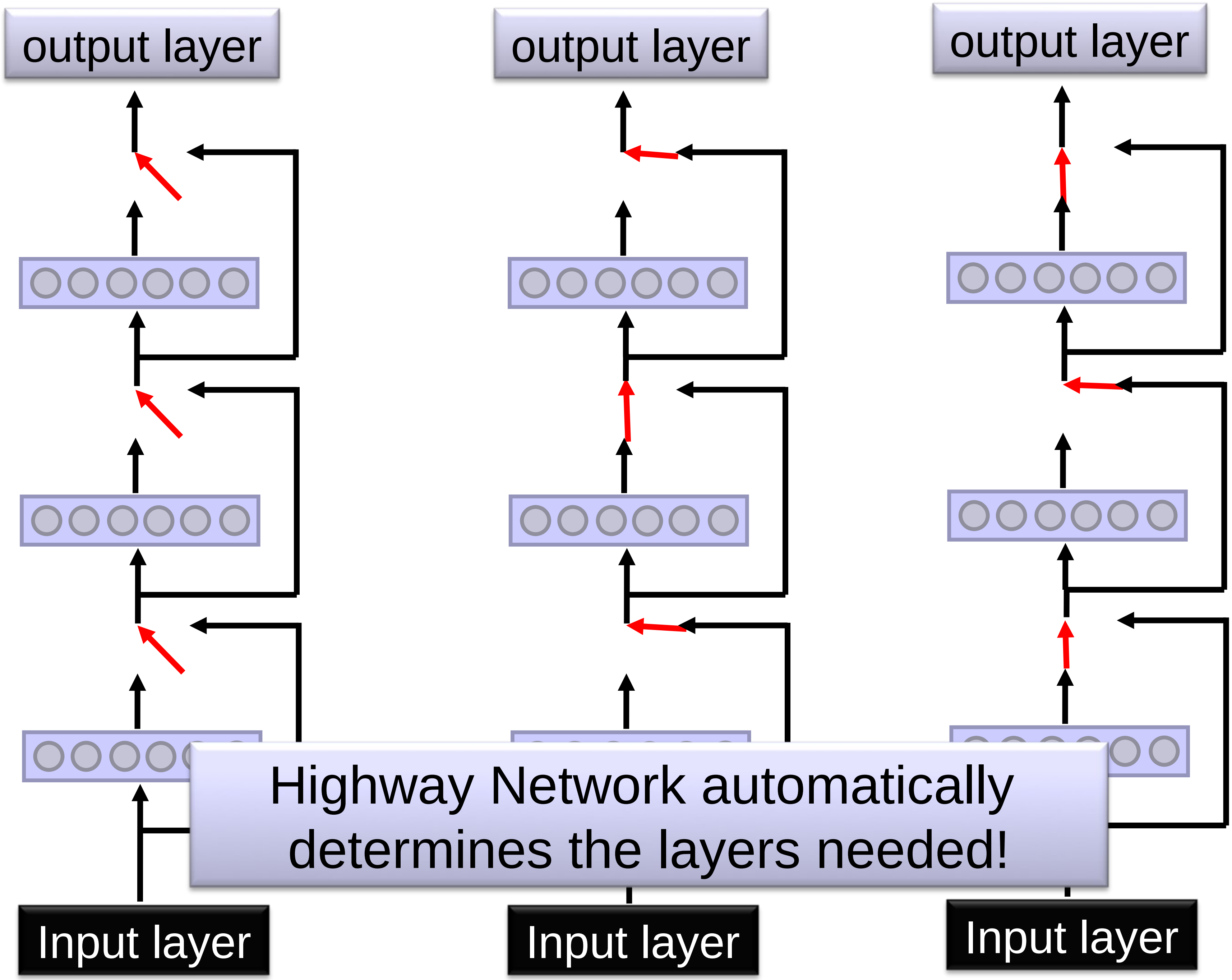


Training Very Deep Networks https://arxiv.org/pdf/1507.06228v2.pdf

- **Residual Network**

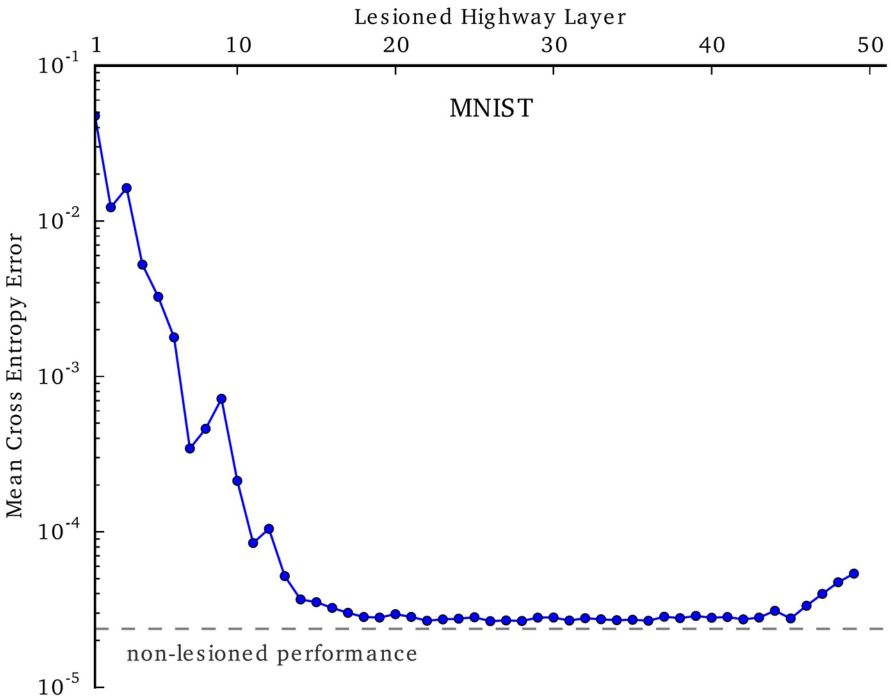


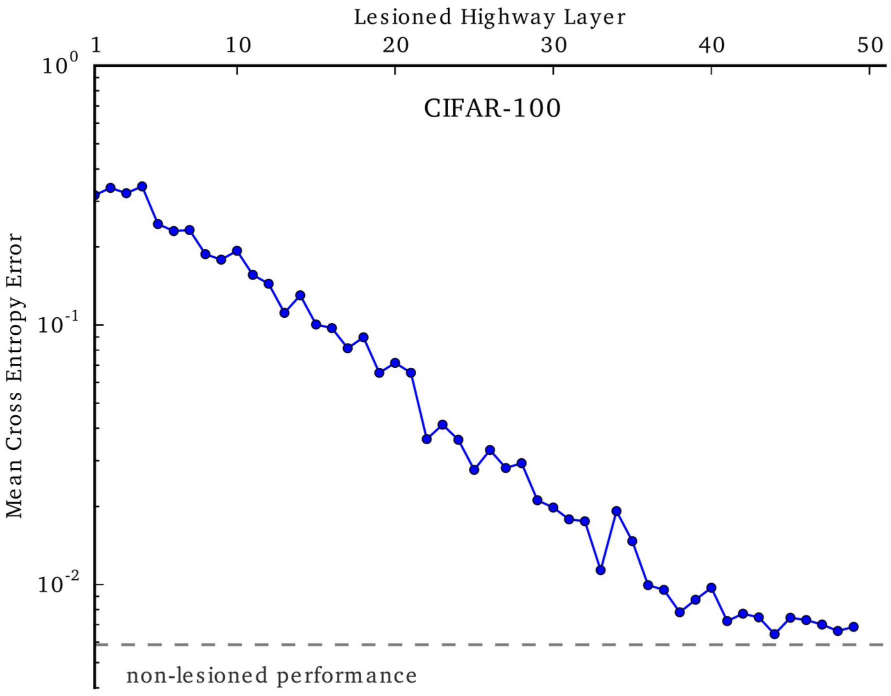
Deep Residual Learning for Image Recognition http://arxiv.org/abs/1512.03385



Highway Network automatically determines the layers needed!

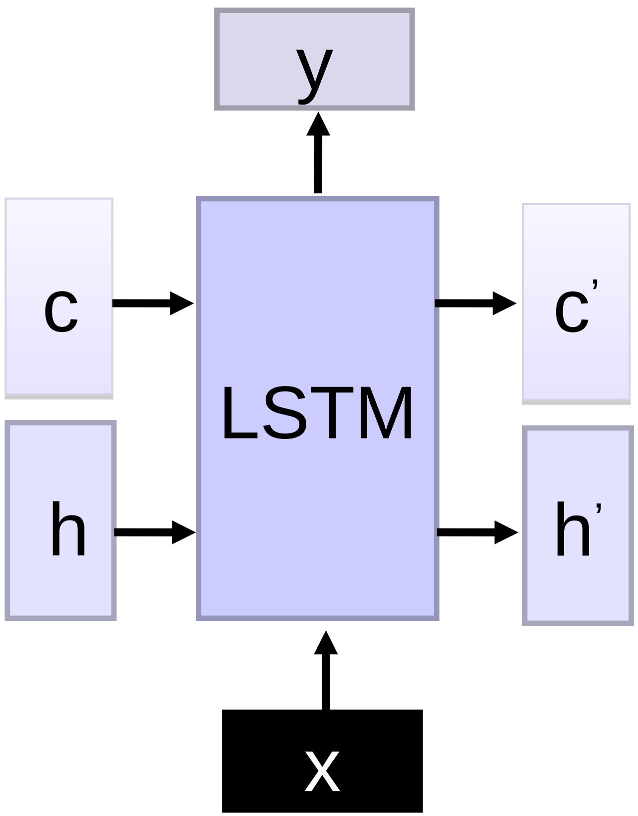
Highway Network Experiments

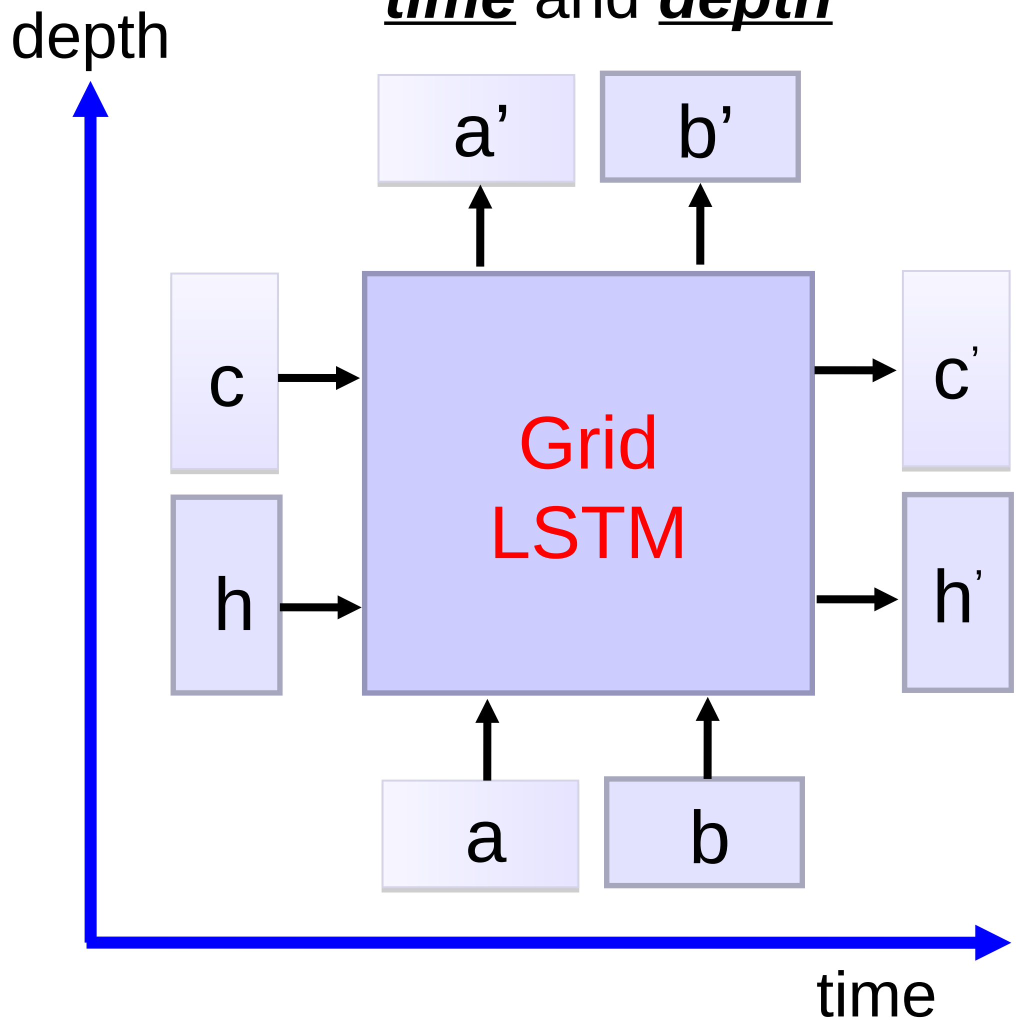




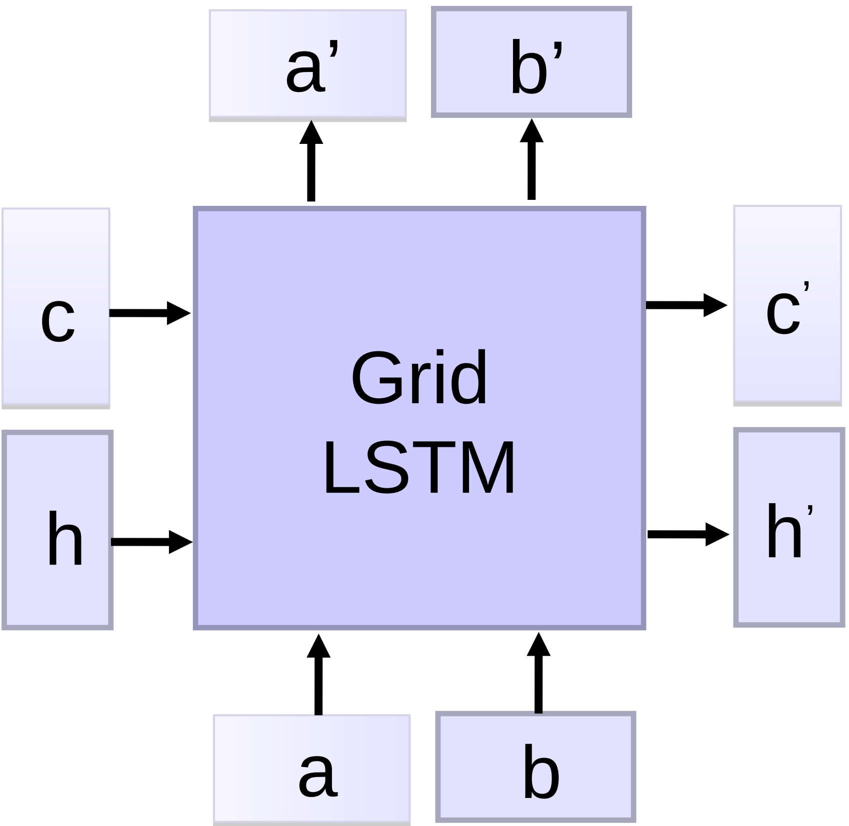
Grid LSTM

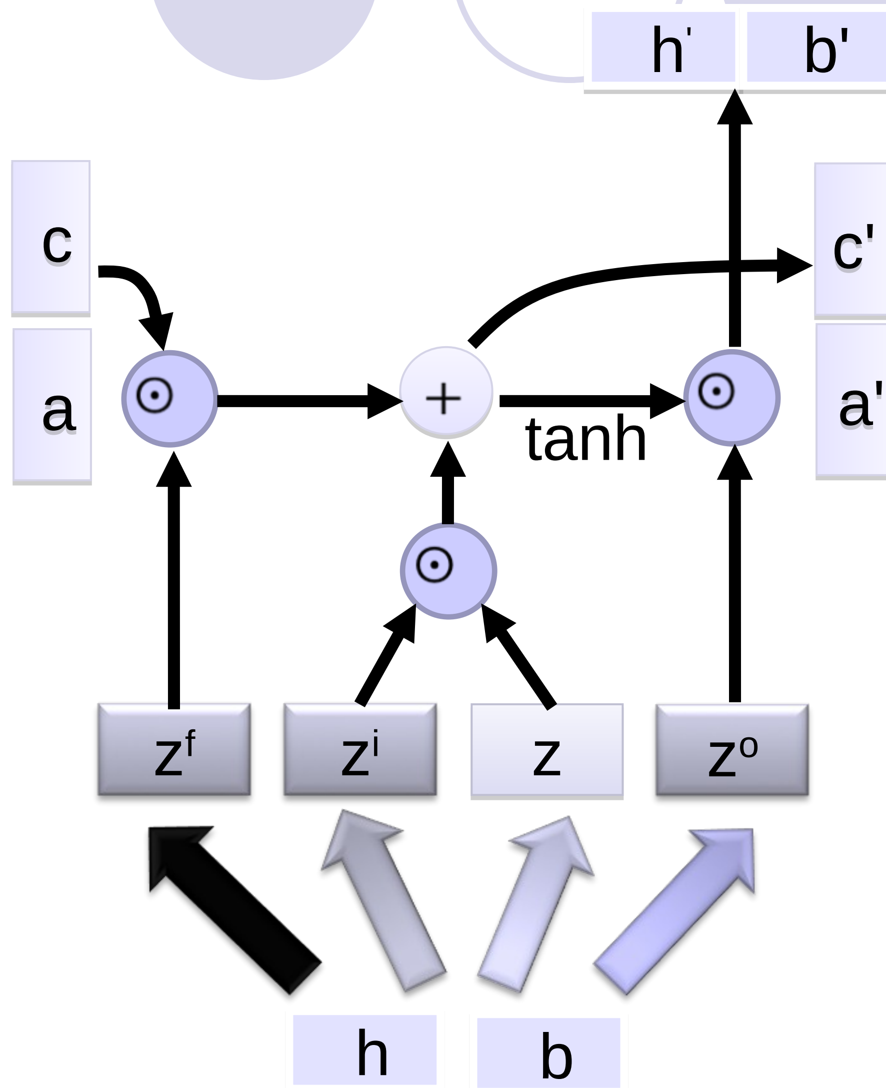
Memory for both ***time*** and ***depth***





Grid LSTM



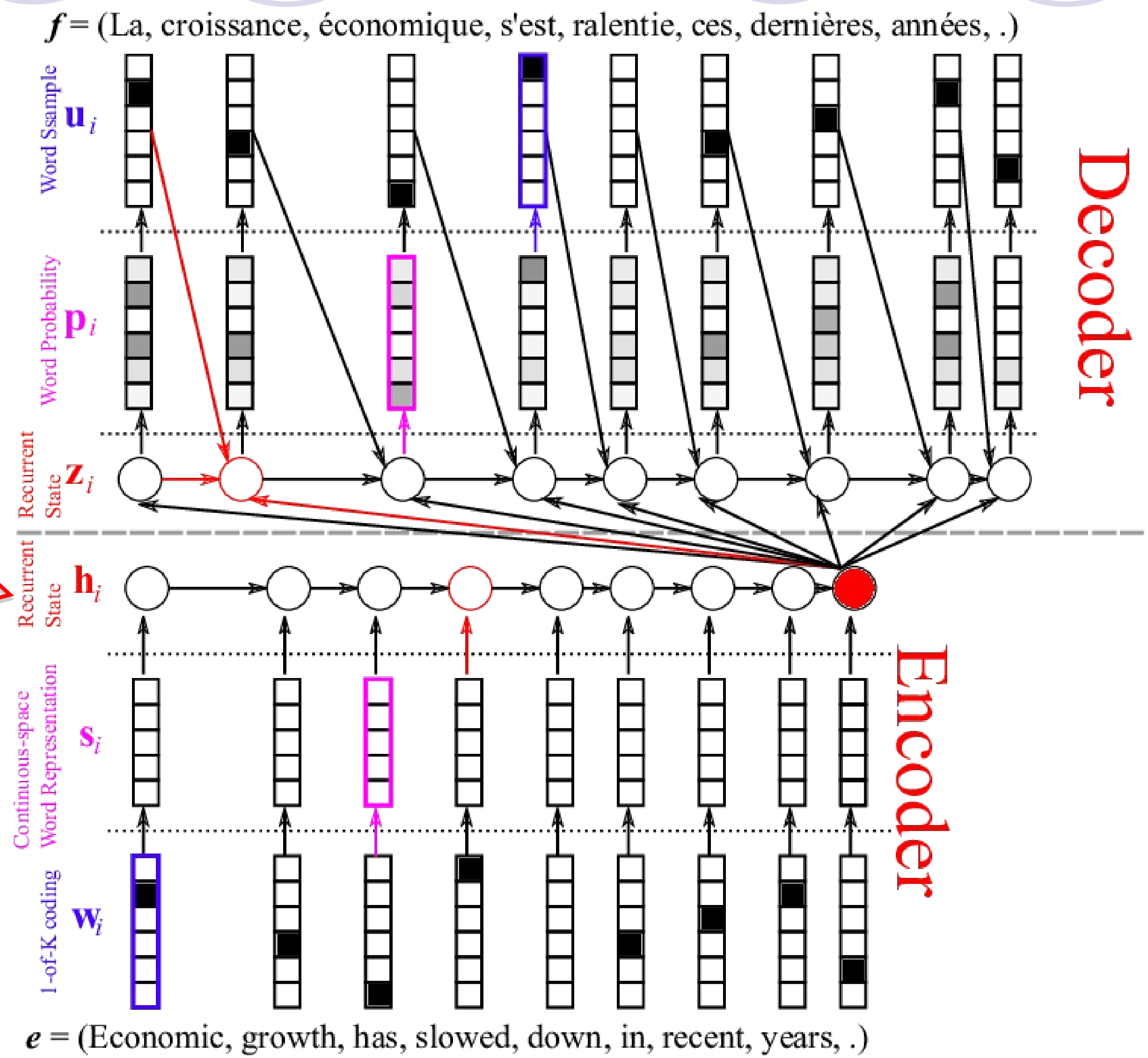


You can generalize this to 3D, and more.

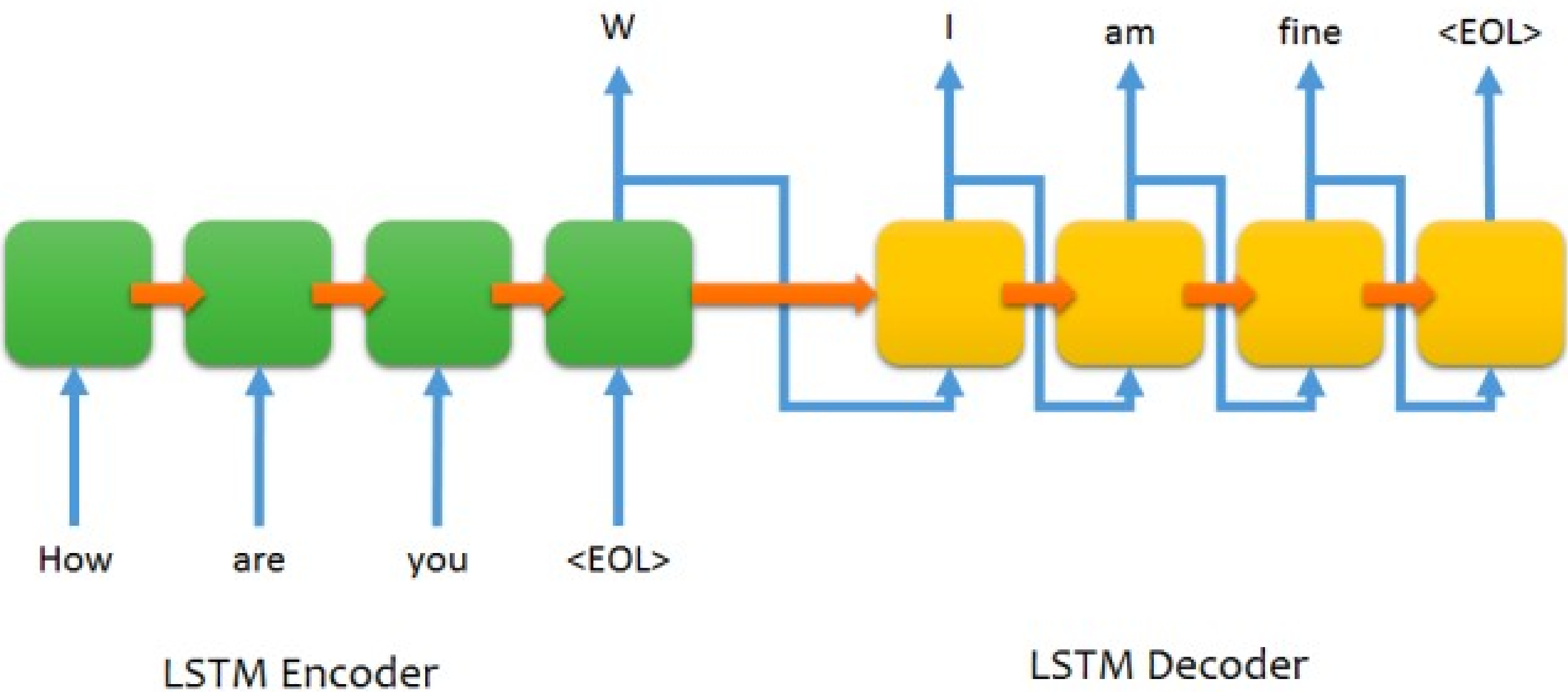
Applications of LSTM / RNN

Neural machine translation

LSTM



Sequence to sequence chat model

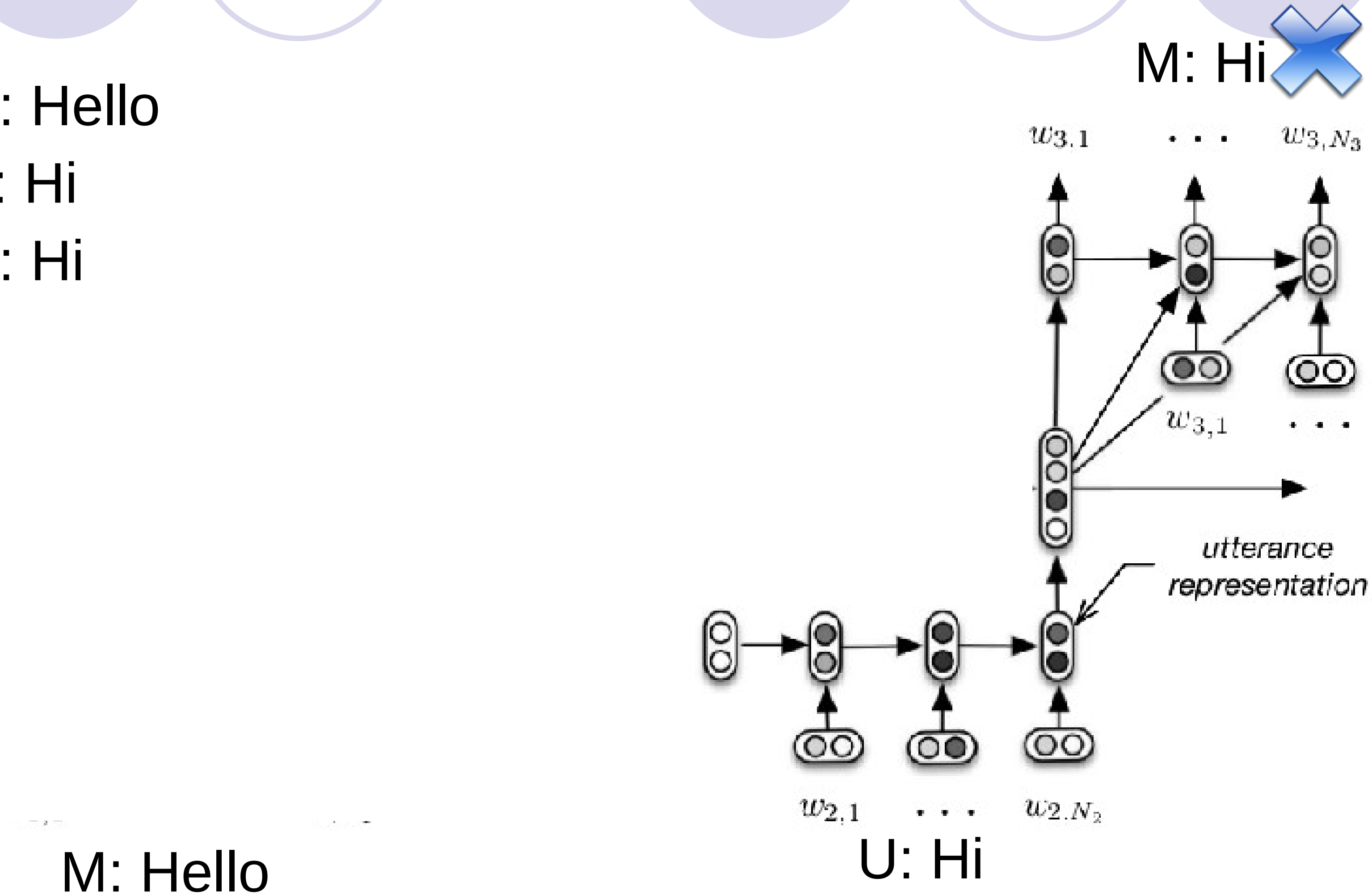


Chat with context

M: Hello

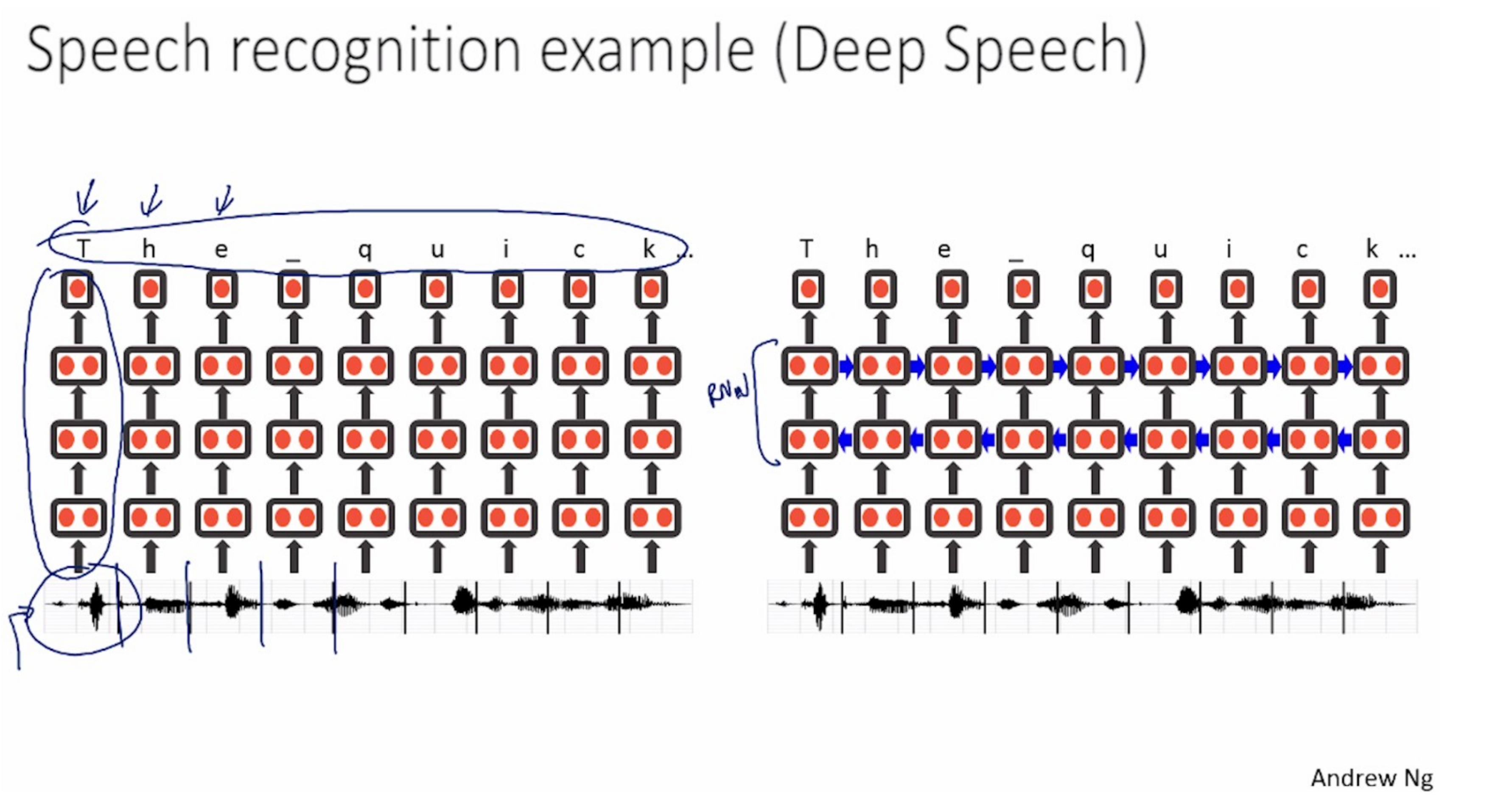
U: Hi

M: Hi

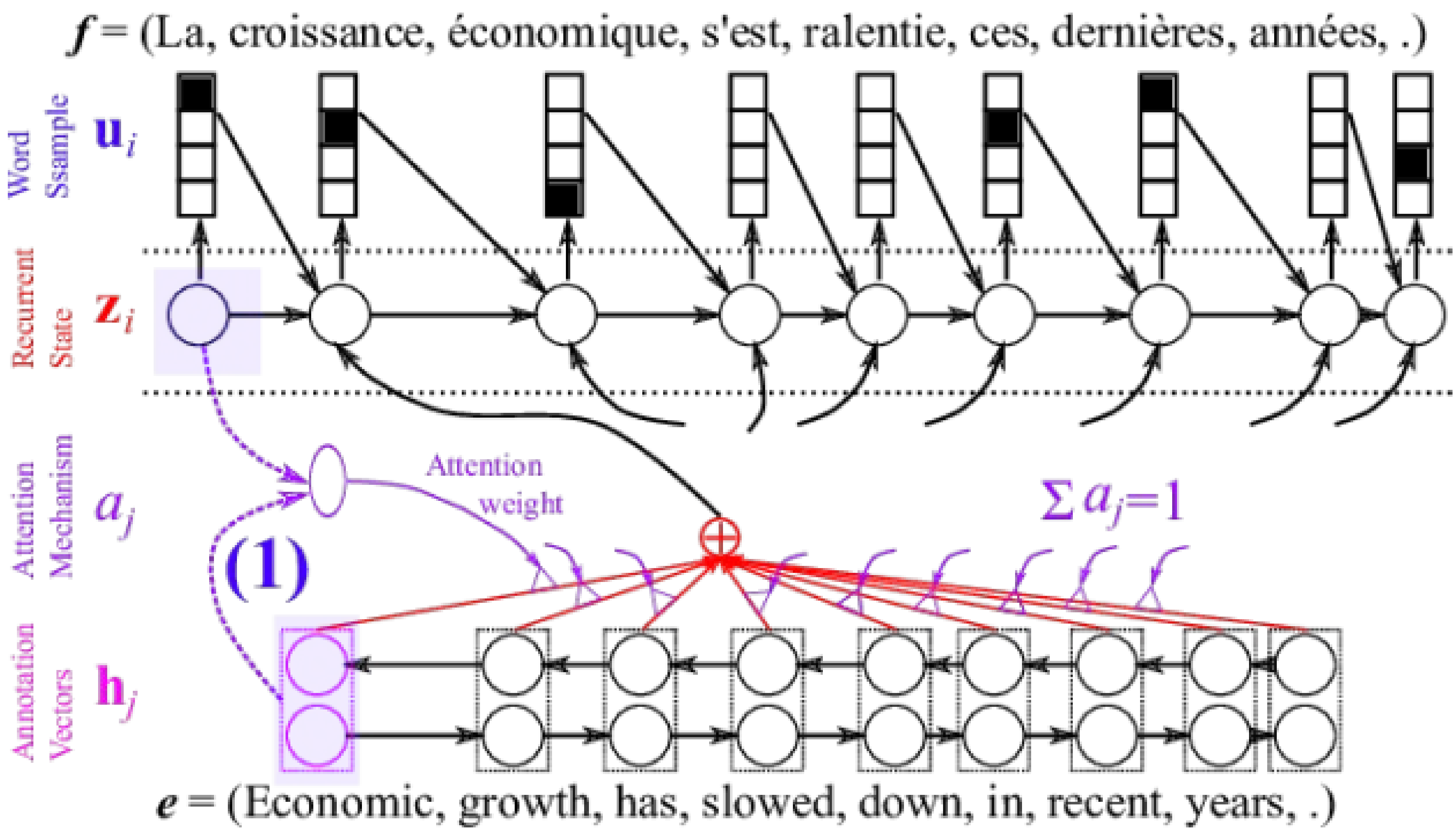


Serban, Iulian V., Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau, 2015 "Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models

Baidu's speech recognition using RNN

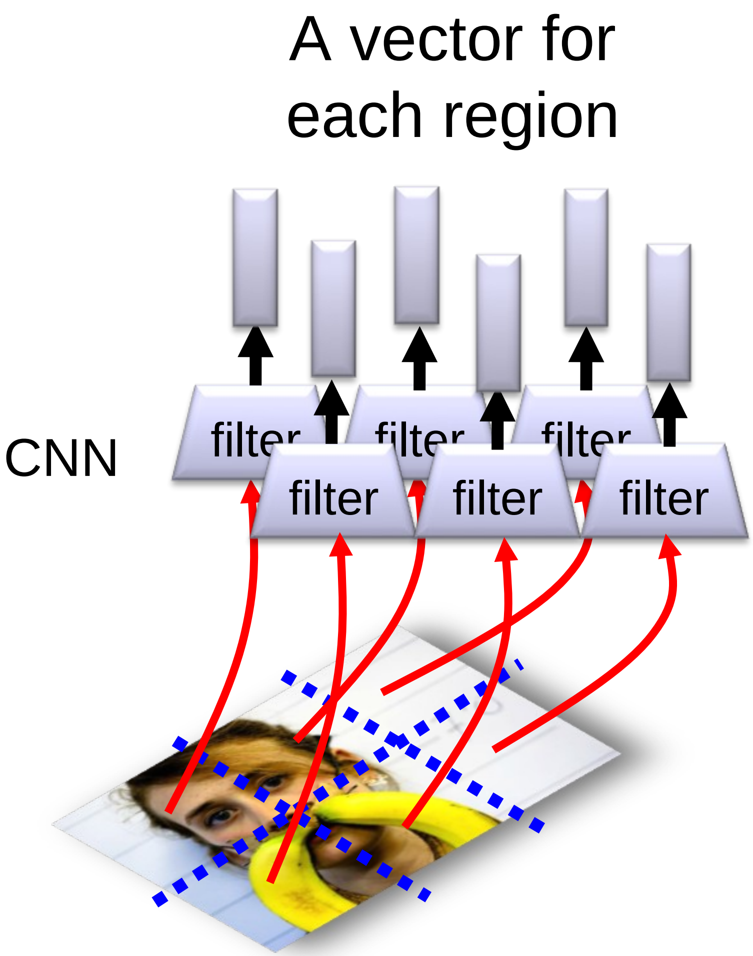


Attention



f = (La, croissance, économique, s'est, ralentie, ces, dernières, années, .)
Word Ssample
$\mathbf{u}_i$
Recurrent State
$\mathbf{z}_i$
Attention Mechanism
a_j
Attention weight
$\Sigma a_j=1$
(1)
Annotation Vectors
$\mathbf{h}_j$
e = (Economic, growth, has, slowed, down, in, recent, years, .)

Image caption generation using attention (From CY Lee lecture)



z^0 is initial parameter, it is also learned

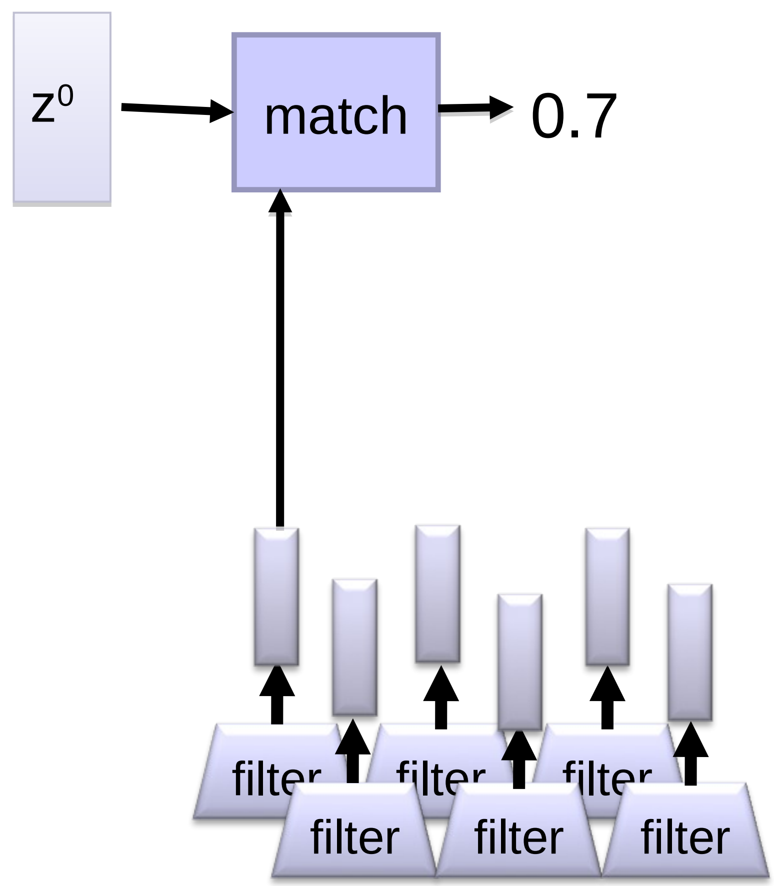


Image Caption Generation

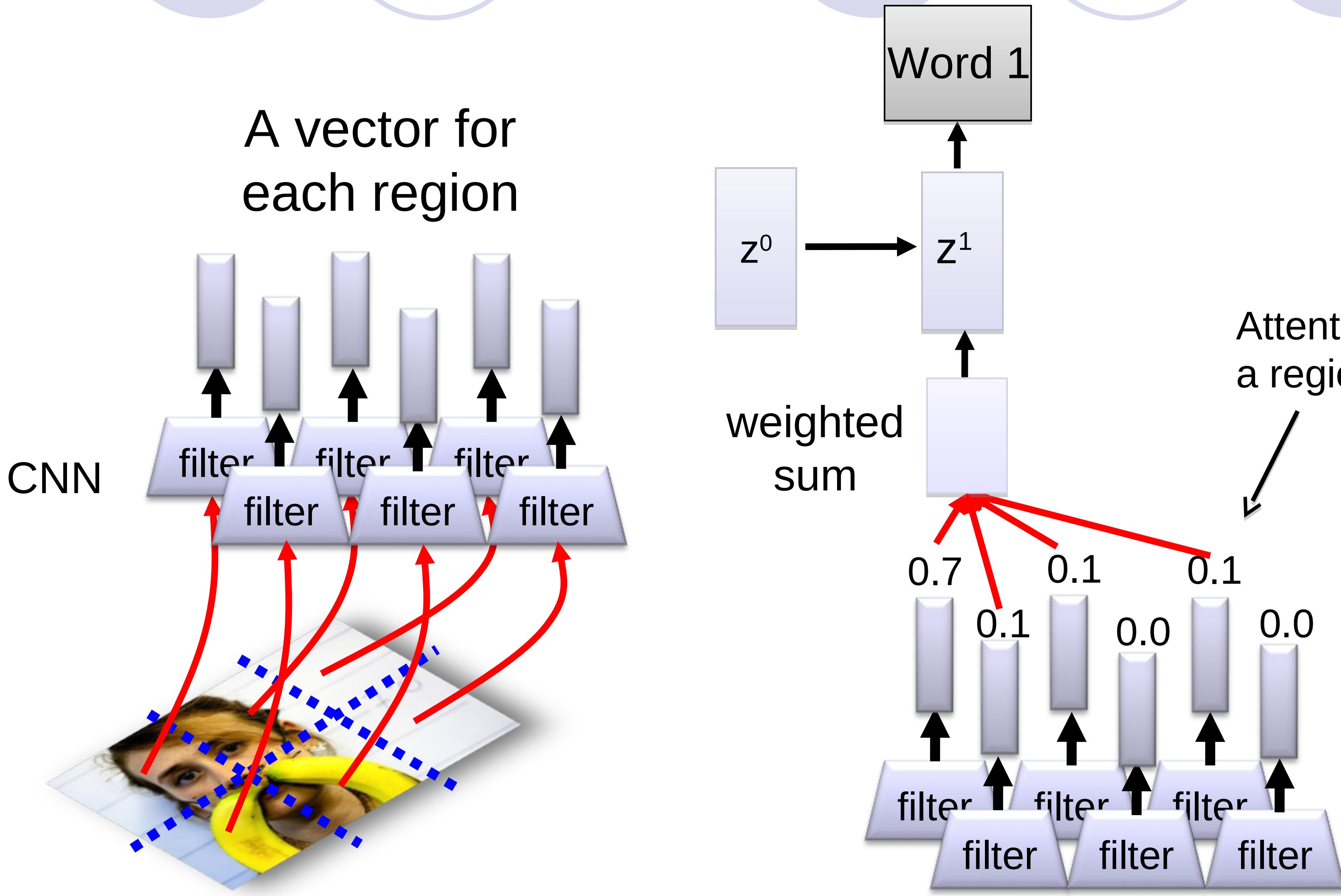
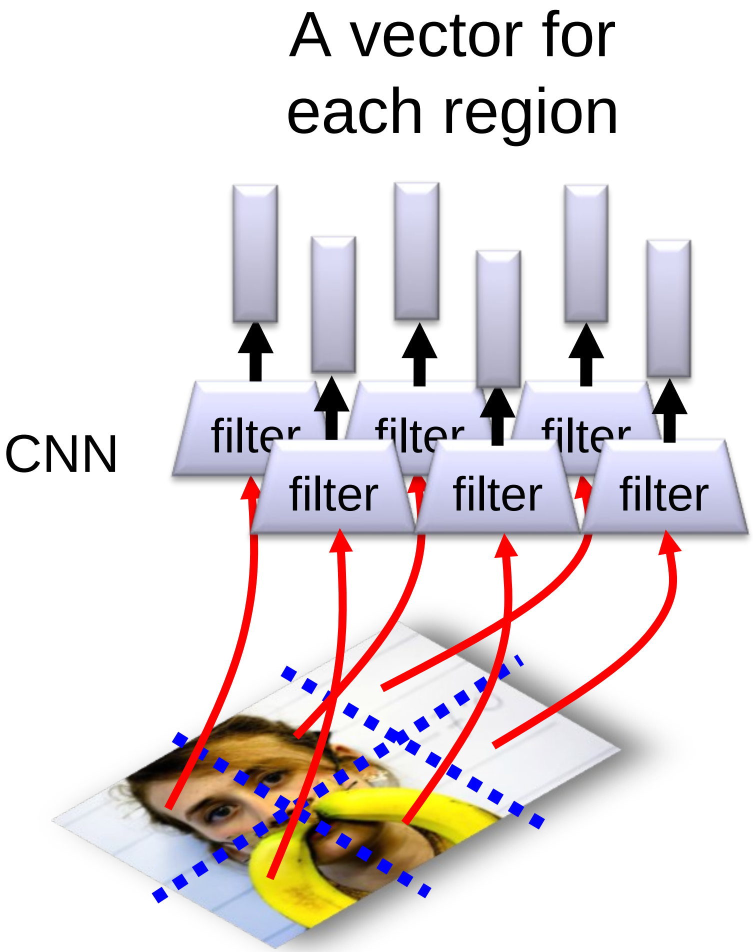


Image Caption Generation



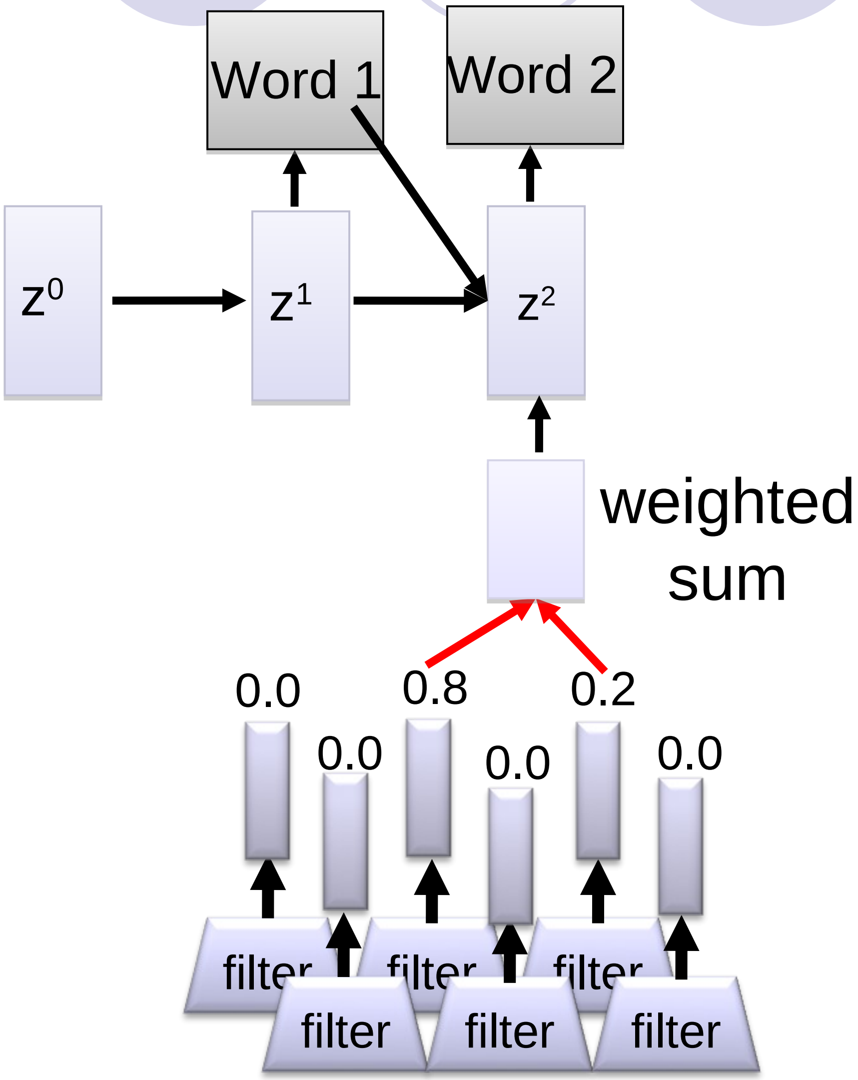
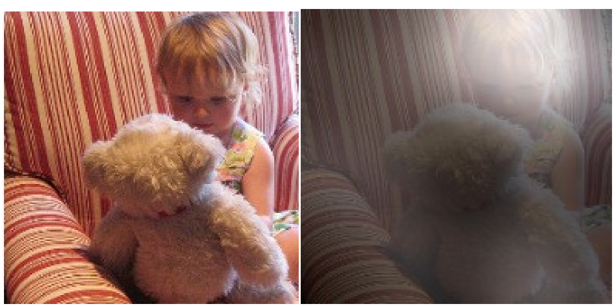


Image Caption Generation

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.



A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML, 2015

Image Caption Generation

A large white bird standing in a forest.

A woman holding a clock in her hand.

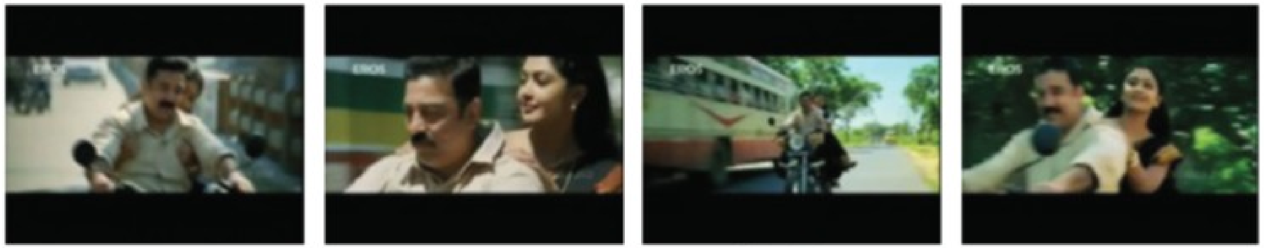
A man wearing a hat and
a hat on a skateboard.

A person is standing on a beach
with a surfboard.

A woman is sitting at a table
with a large pizza.

A man is talking on his cell phone
while another man watches.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML, 2015



Ref: A man and a woman ride a motorcycle

A **man** and a **woman** are **talking** on the **road**

* Possible project?

Ref: A woman is frying food

Someone is **frying** a **fish** in a **pot**

Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, Aaron Courville, “Describing Videos by Exploiting Temporal Structure”, ICCV, 2015