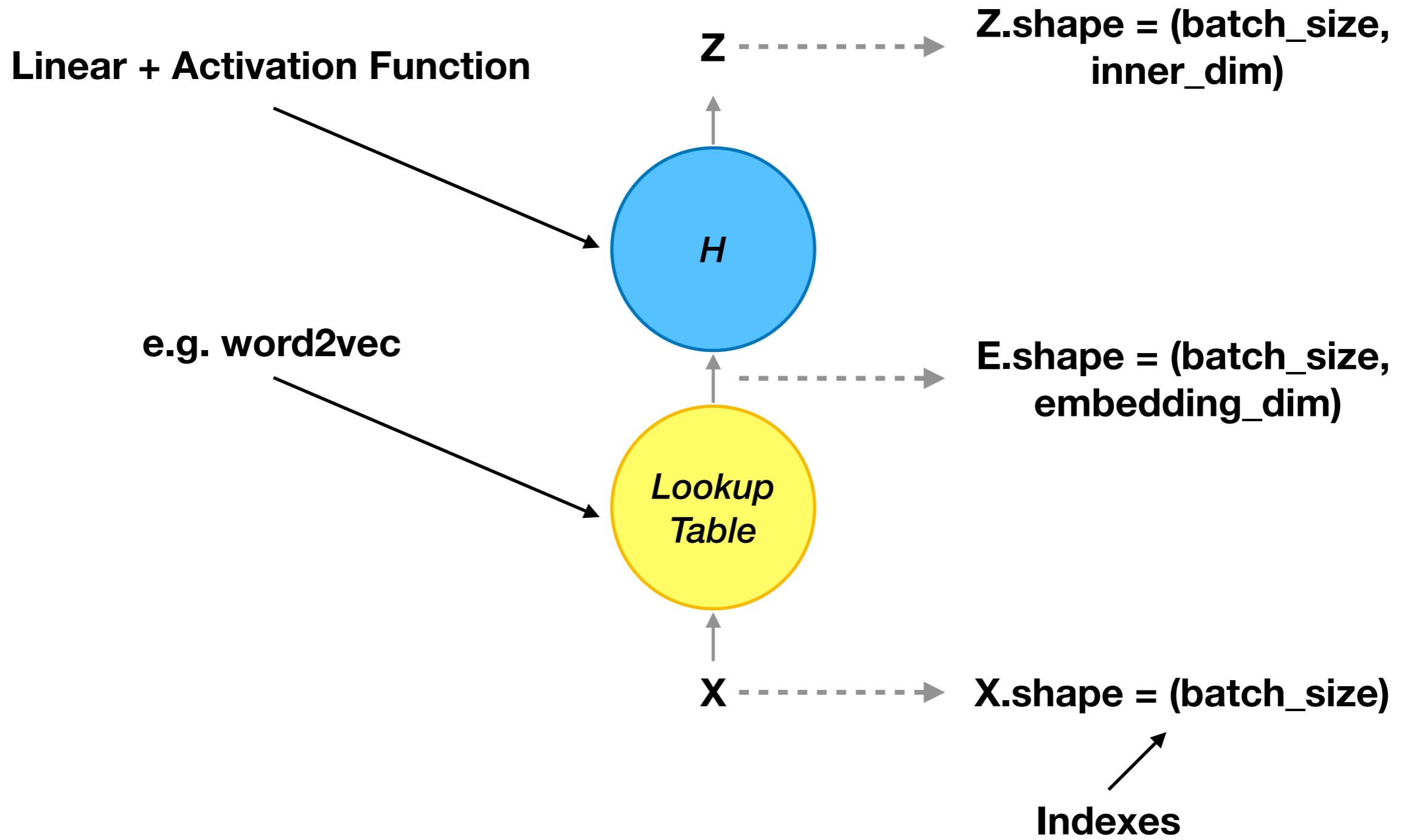


Recurrent Neural Network

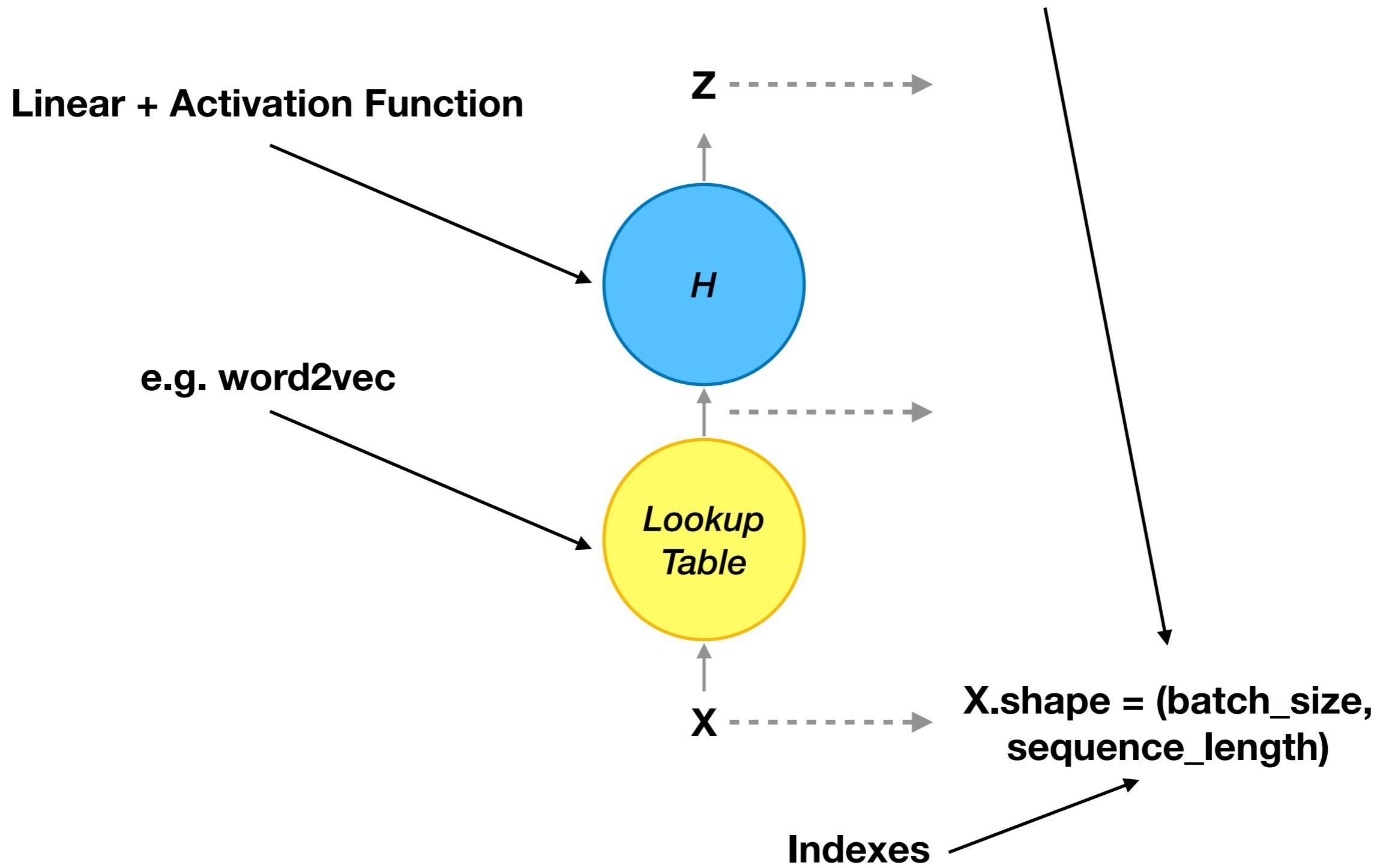
Materials MS DL Course of Boris Zubarev @bobazooba

Neural Network



Neural Network

What if we have different lengths?



Padding

[Мама, мыла, раму]

[Покупать, новый, телефон, затратно]

[Bay]

[Мне, очень, не, нравится ваш, банк]



max_sequence_length = 8

Depend of your dataset statistics



[Мама, мыла, раму, PAD, PAD, PAD, PAD, PAD]

[Покупать, новый, телефон, затратно, PAD, PAD, PAD, PAD]

[Bay, PAD, PAD, PAD, PAD, PAD, PAD]

[Мне, очень, не, нравится, ваш, банк, PAD, PAD]

Padding

[Мама, мыла, раму, PAD, PAD, PAD, PAD, PAD]

[Покупать, новый, телефон, затратно, PAD, PAD, PAD, PAD]

[Bay, PAD, PAD, PAD, PAD, PAD, PAD]

[Мне, очень, не, нравится, ваш, банк, PAD, PAD]

Indexing

Collect word2index vocabulary

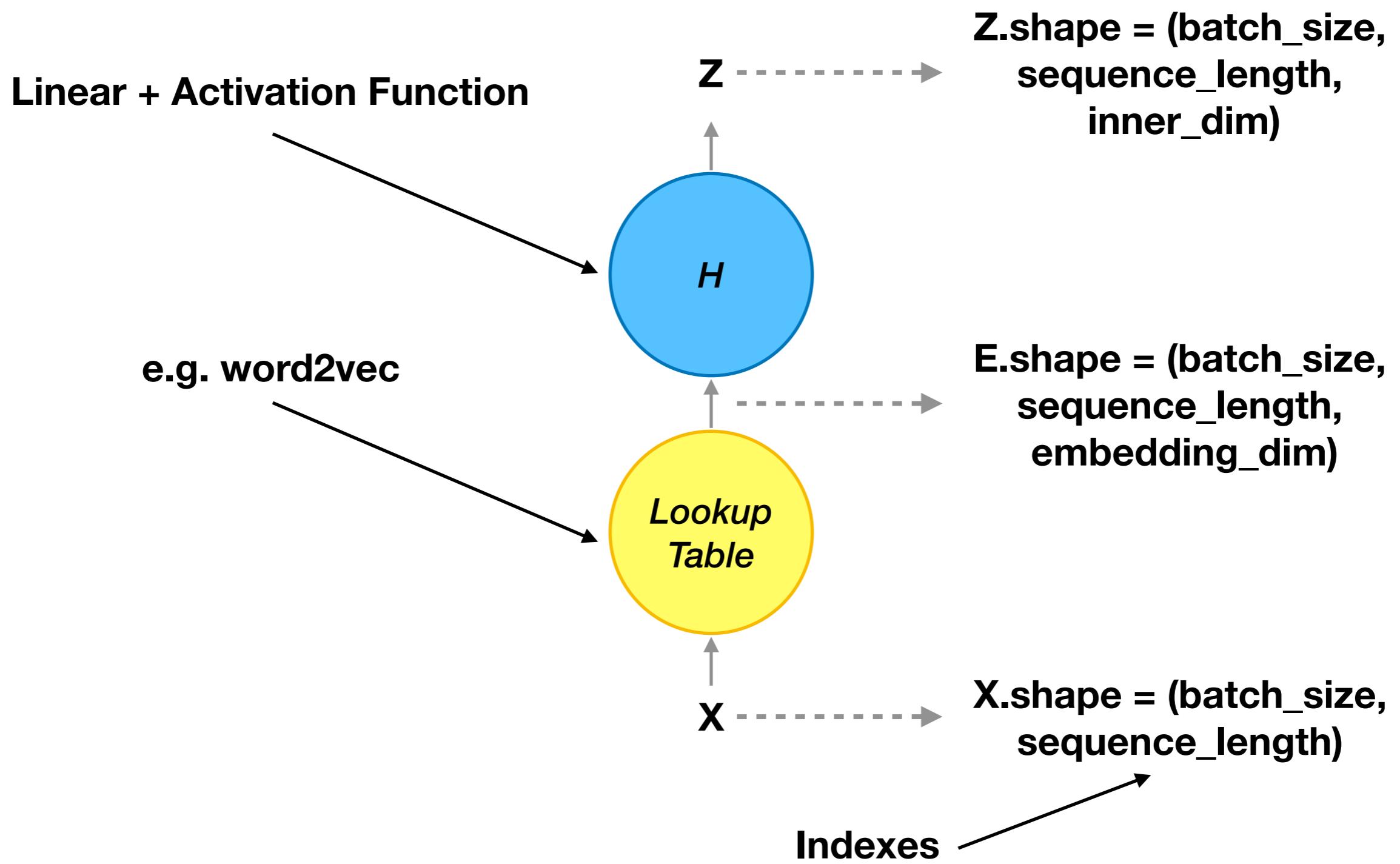
34 2 3 0 0 0 0 0

64 23 1515 45 0 0 0 0

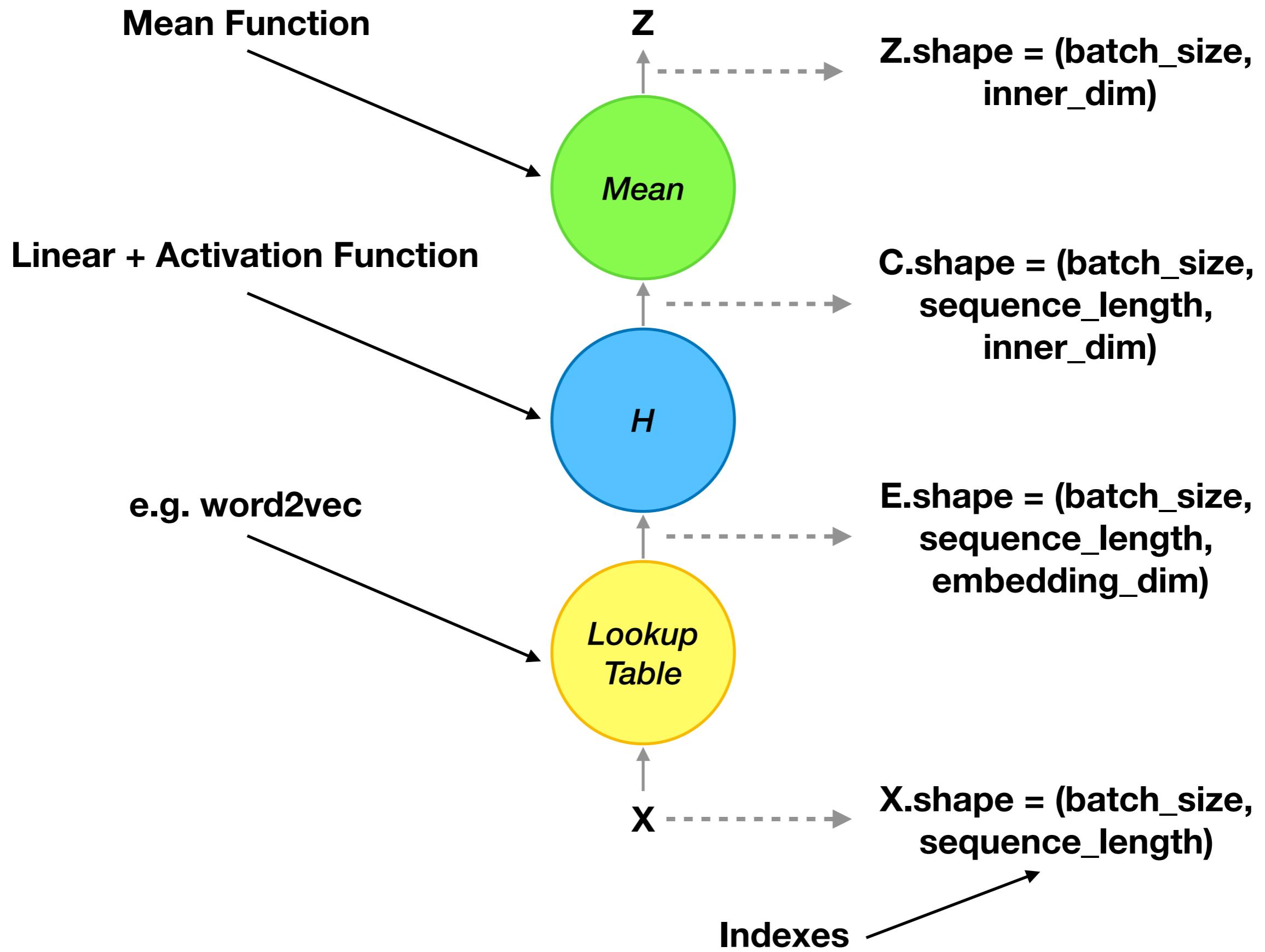
55 0 0 0 0 0 0 0

36 43 22 7 124 356 0 0

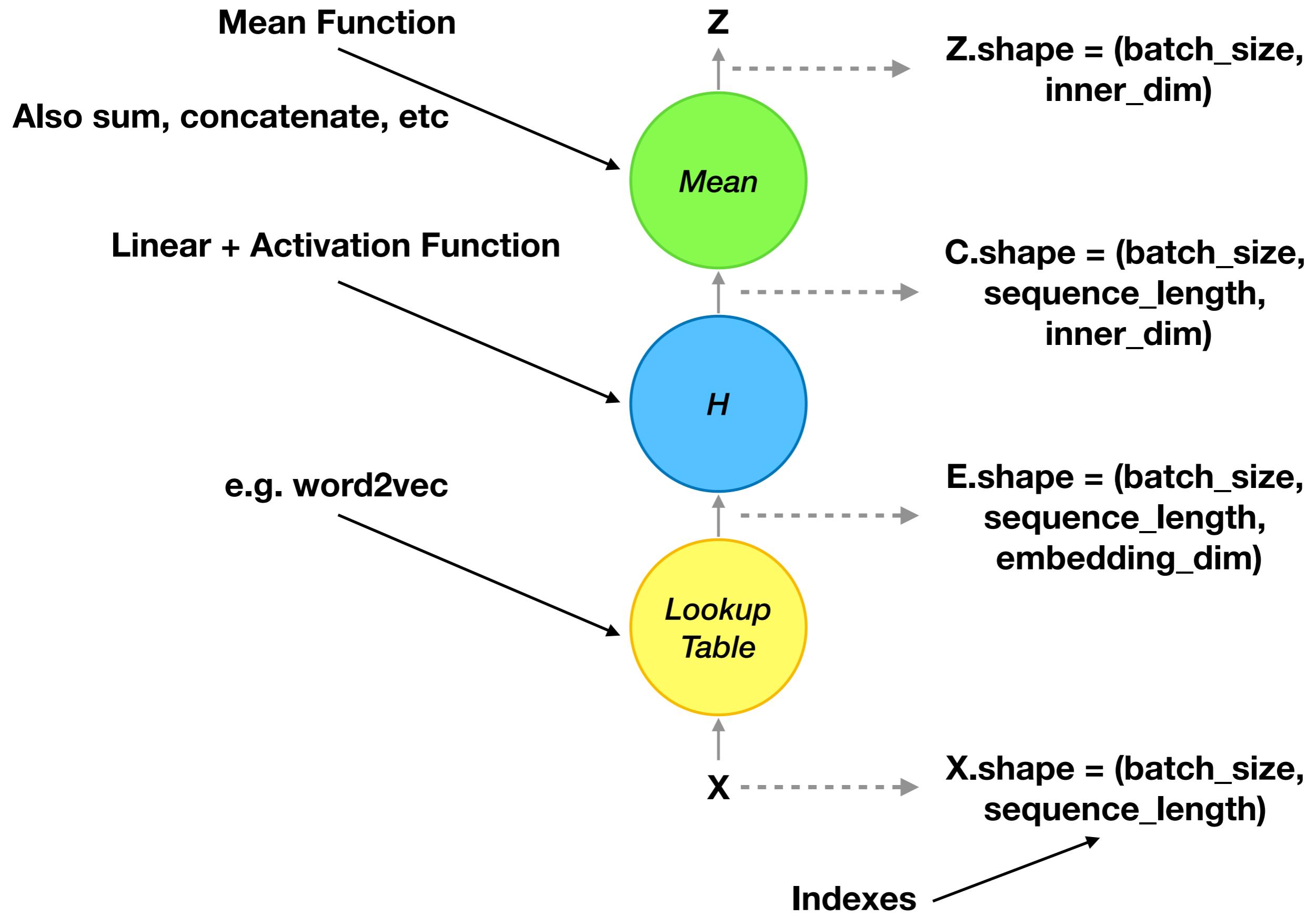
Neural Network



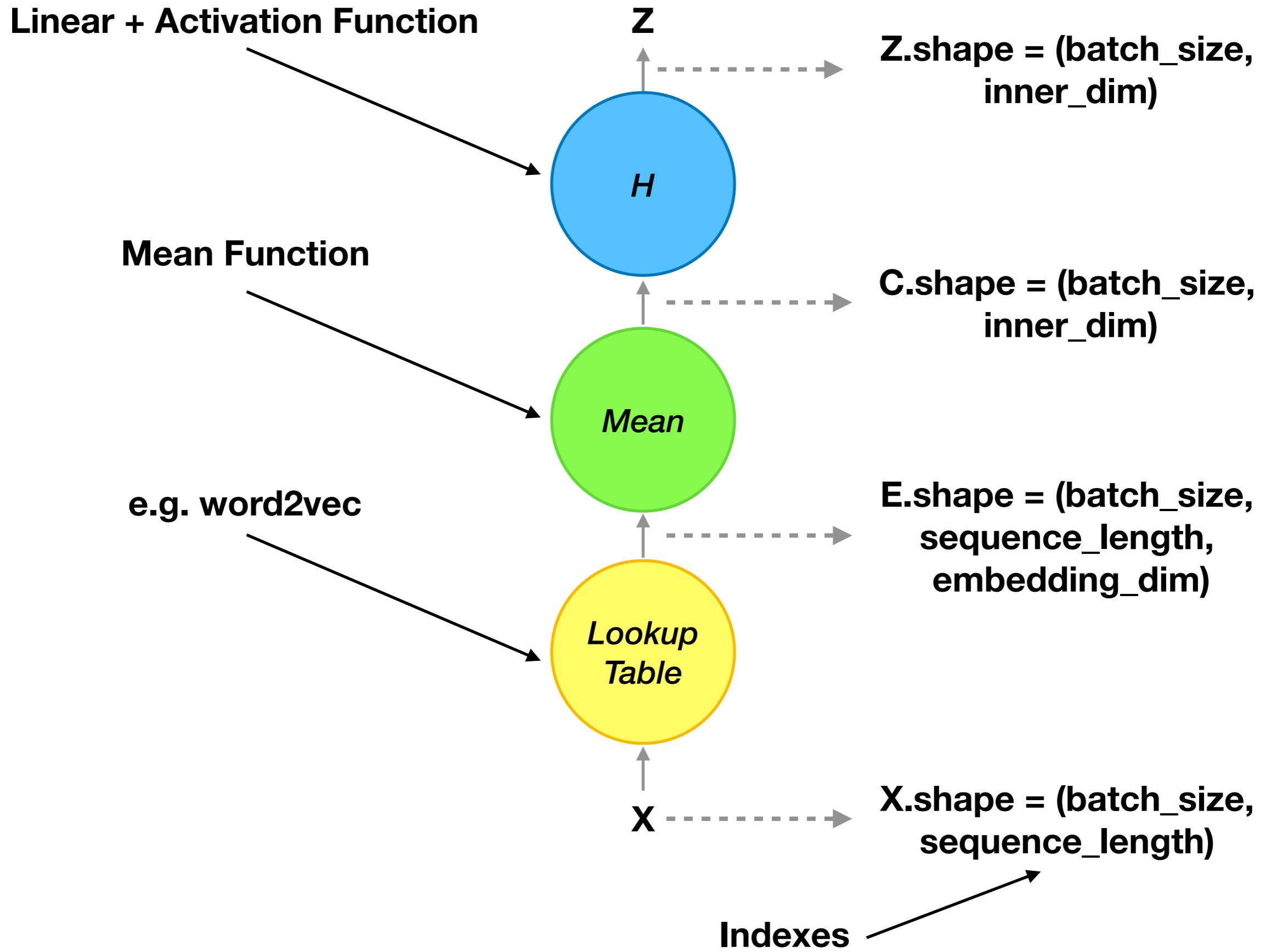
Neural Network



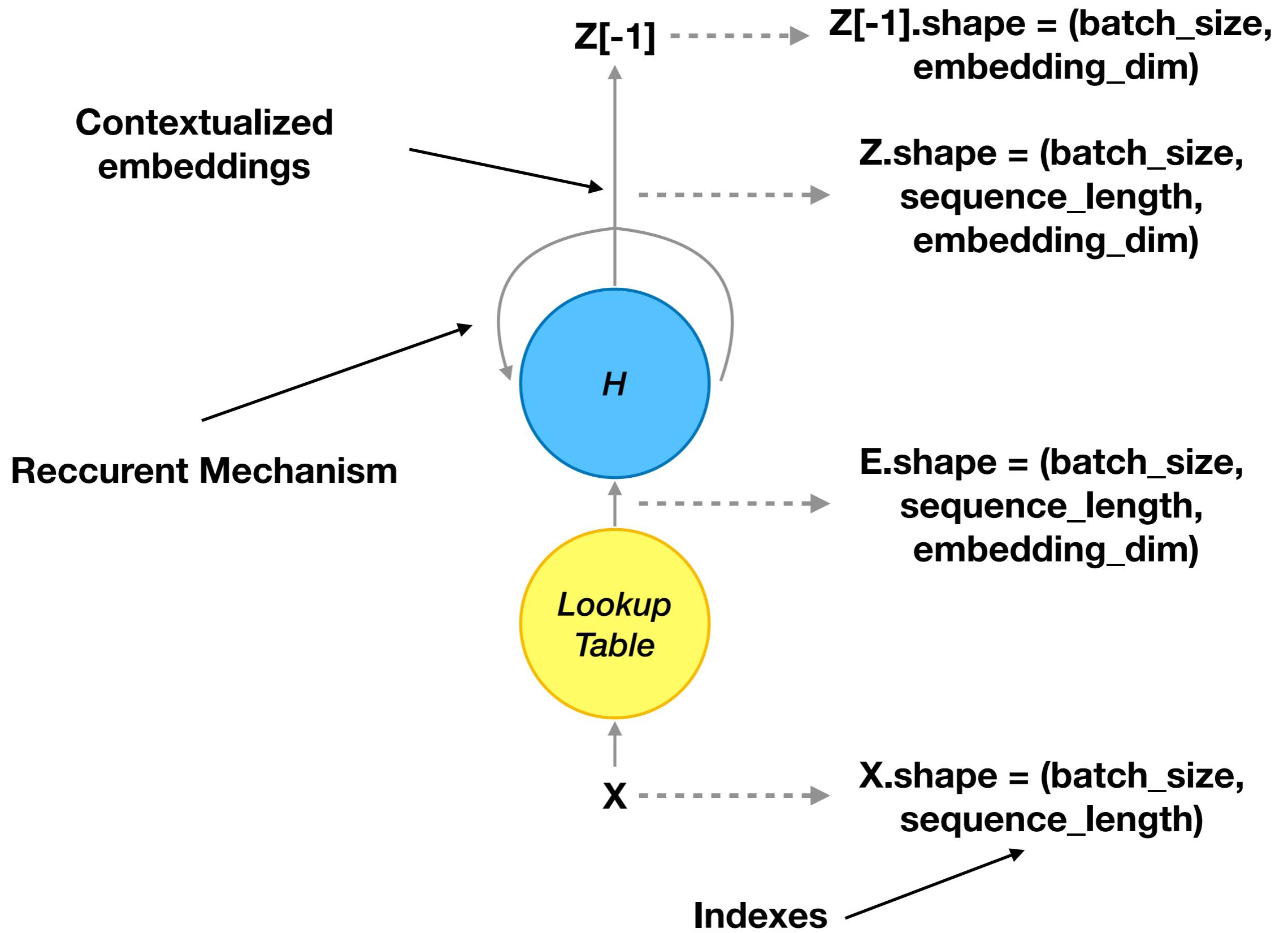
Neural Network



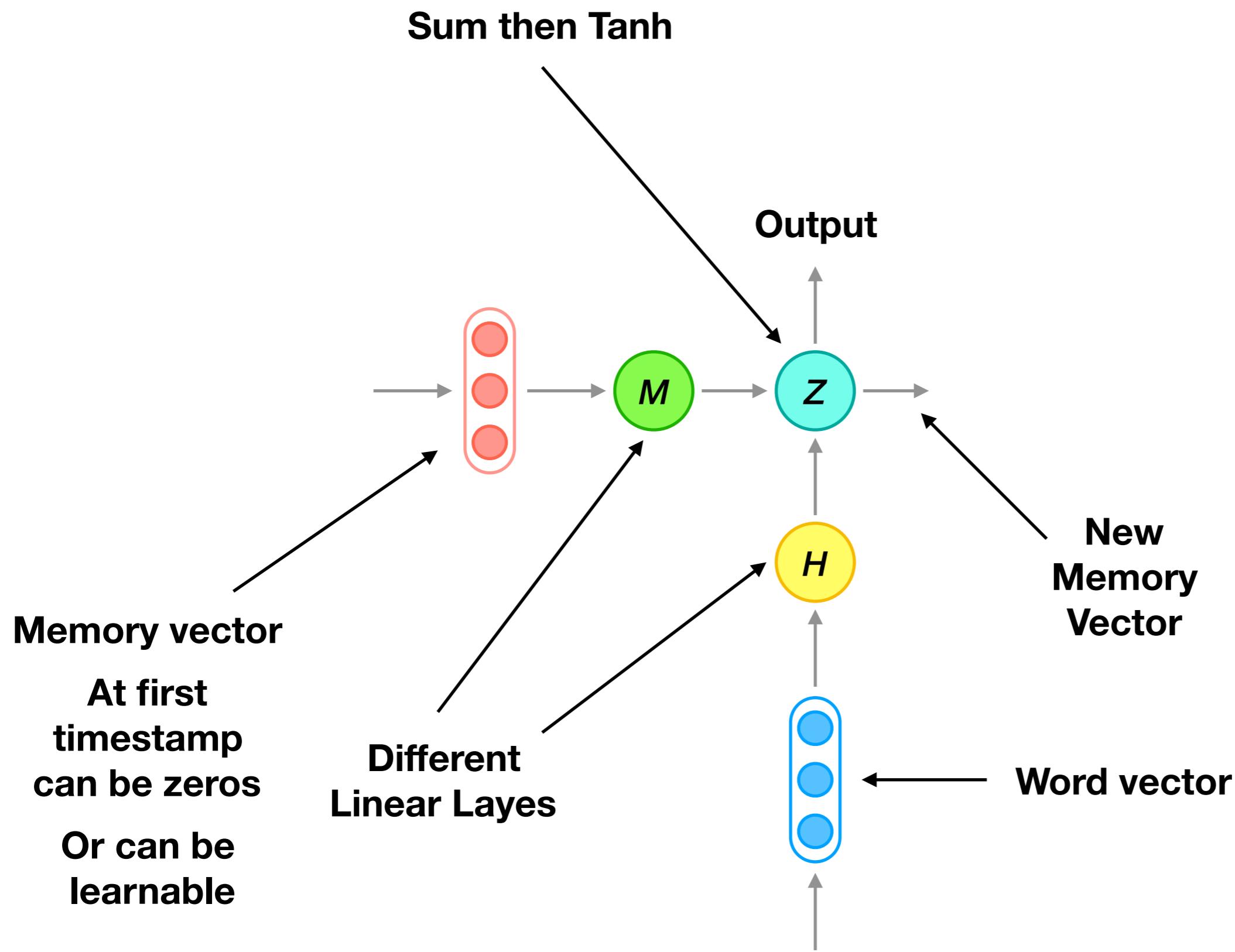
Neural Network



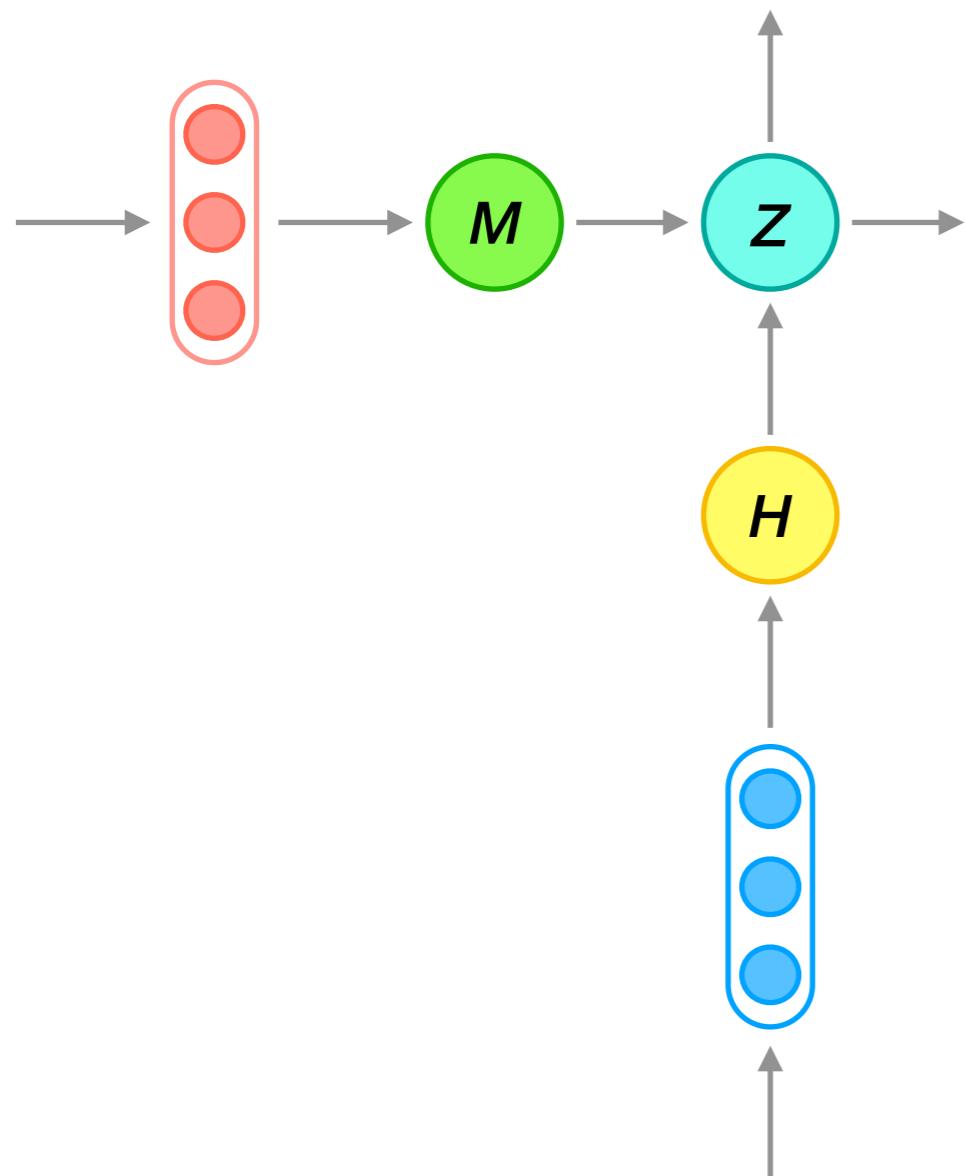
Recurrent Neural Network



Reccurrent Cell



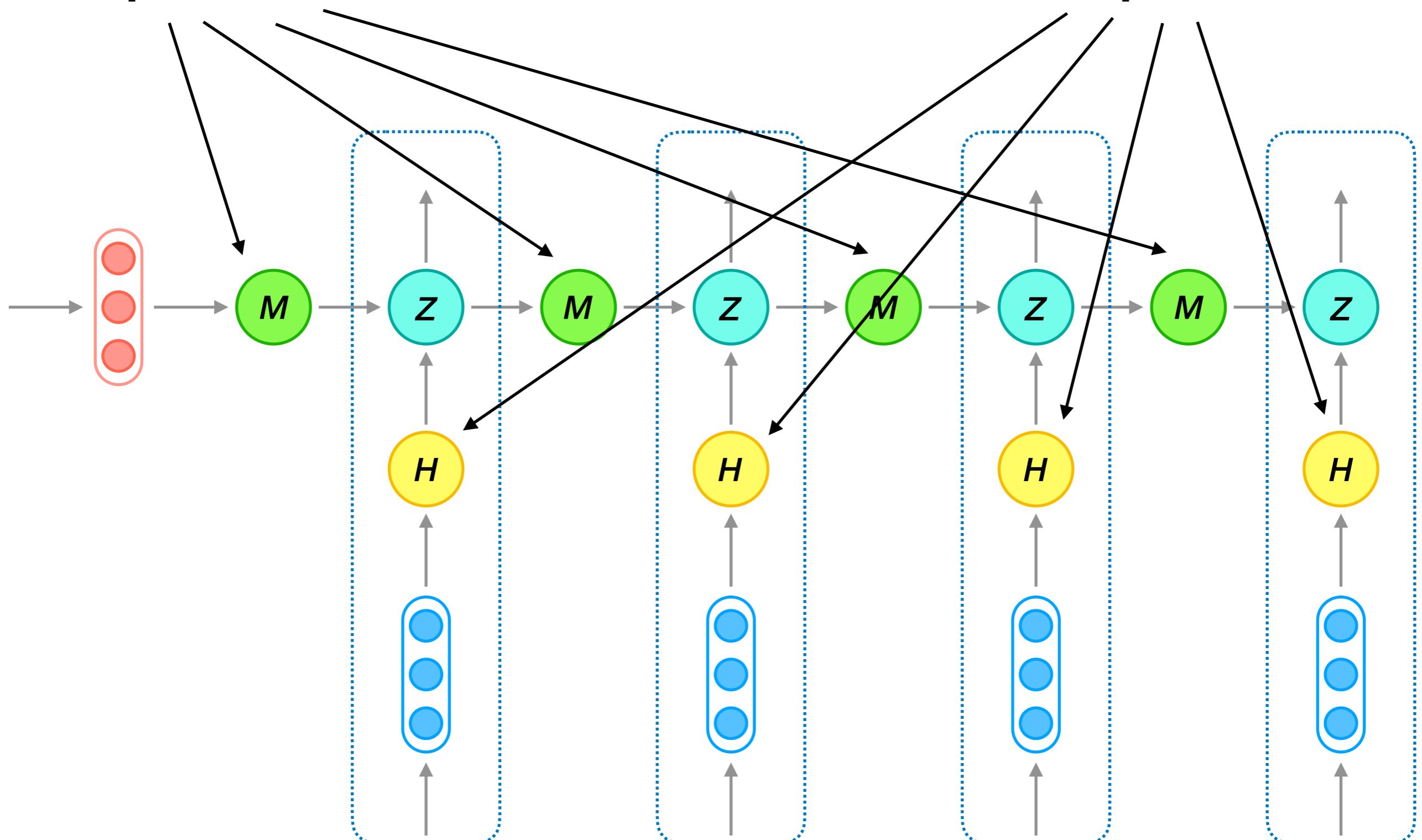
Reccurent Mechanism



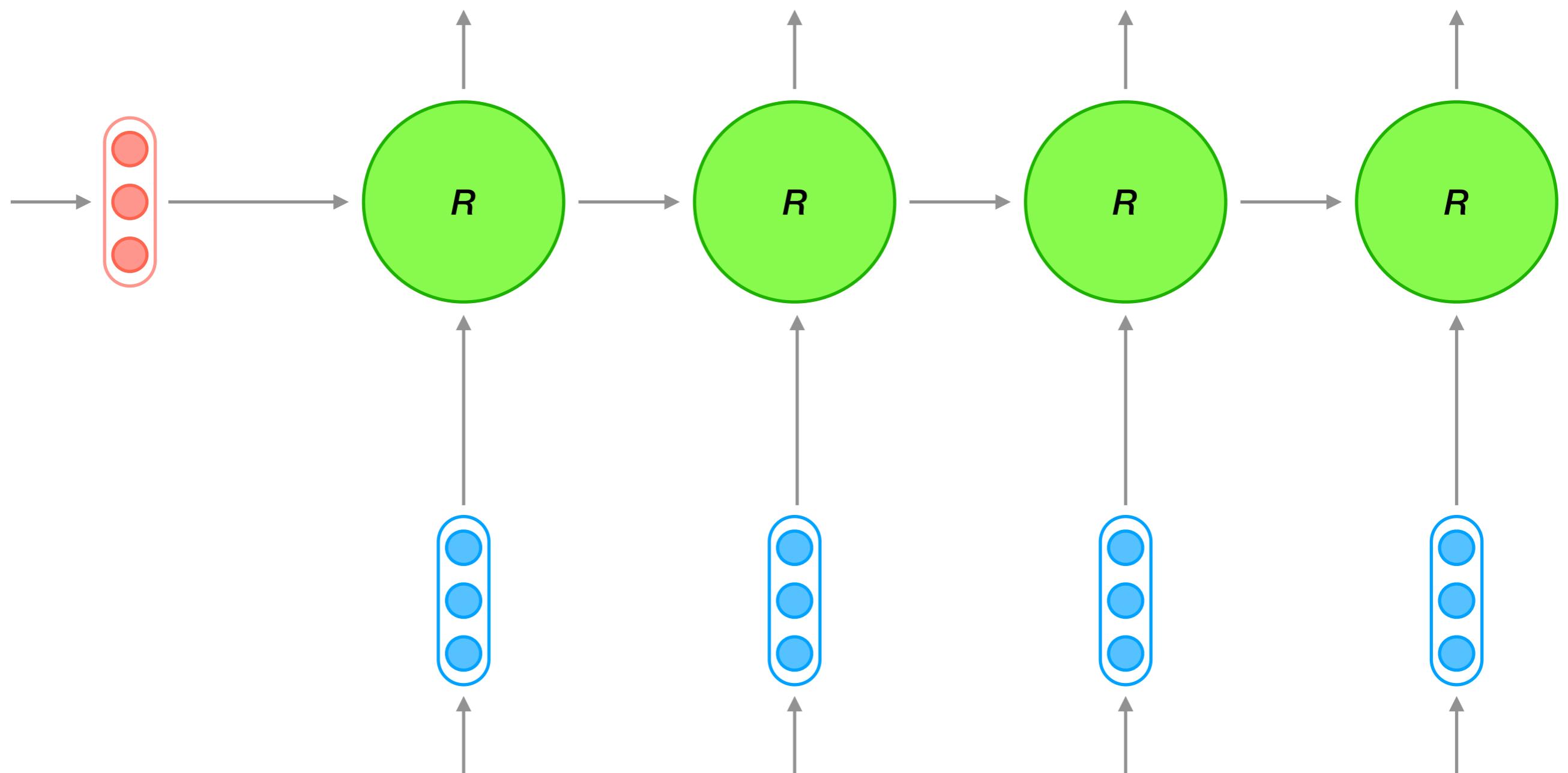
Recurrent Mechanism

Same parameters!

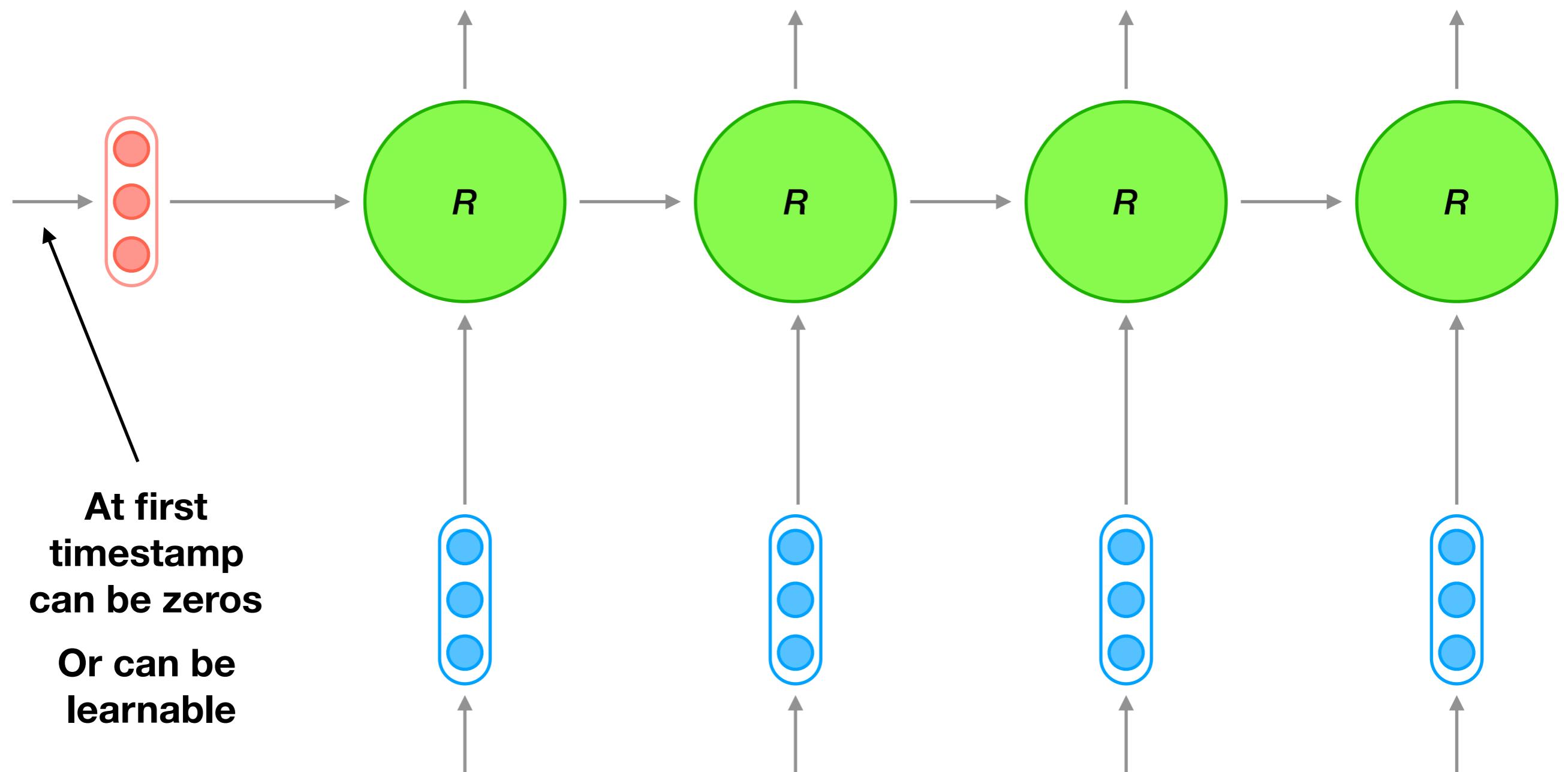
Another
same parameters!



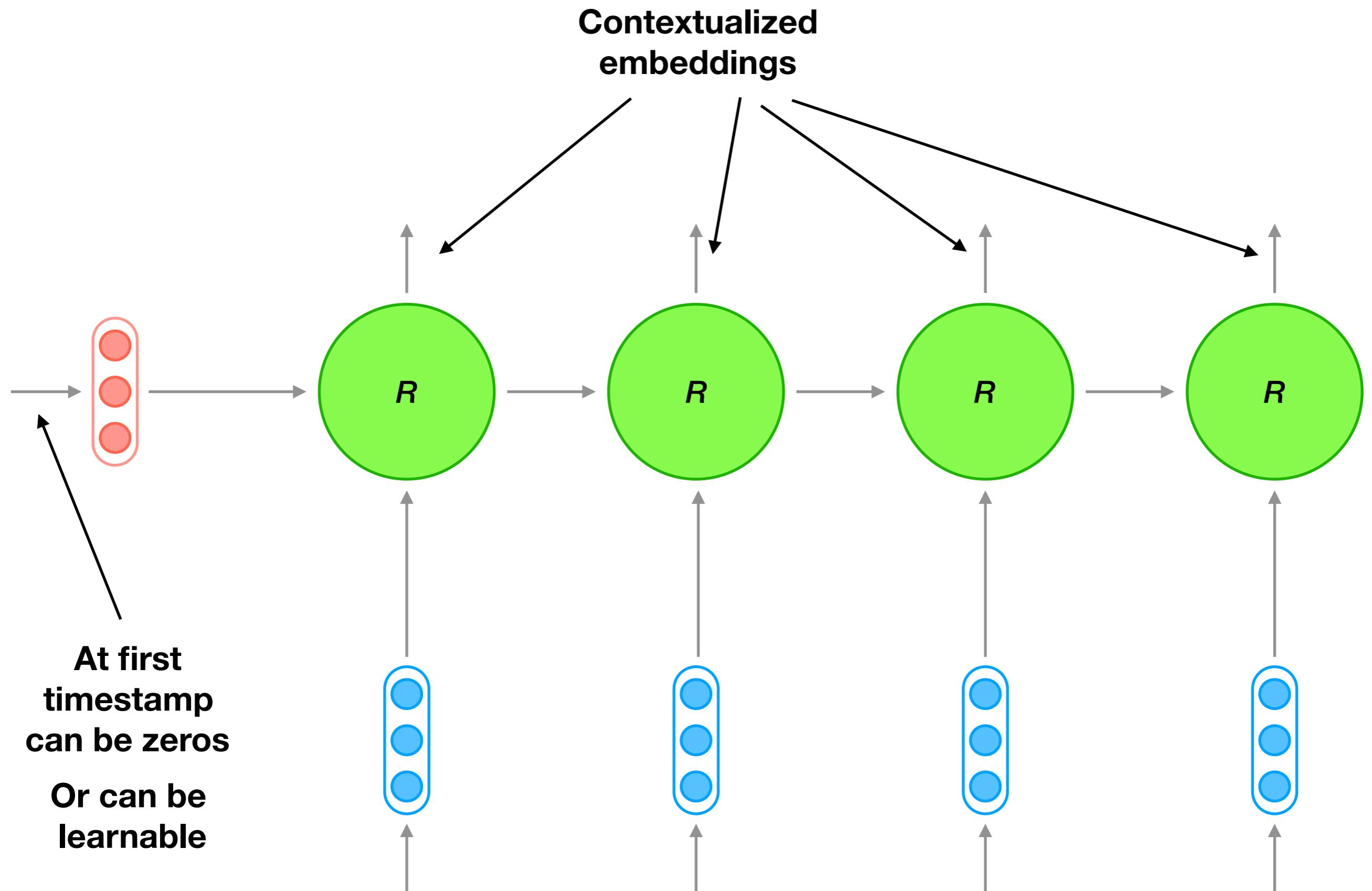
Reccurent Mechanism



Reccurent Mechanism

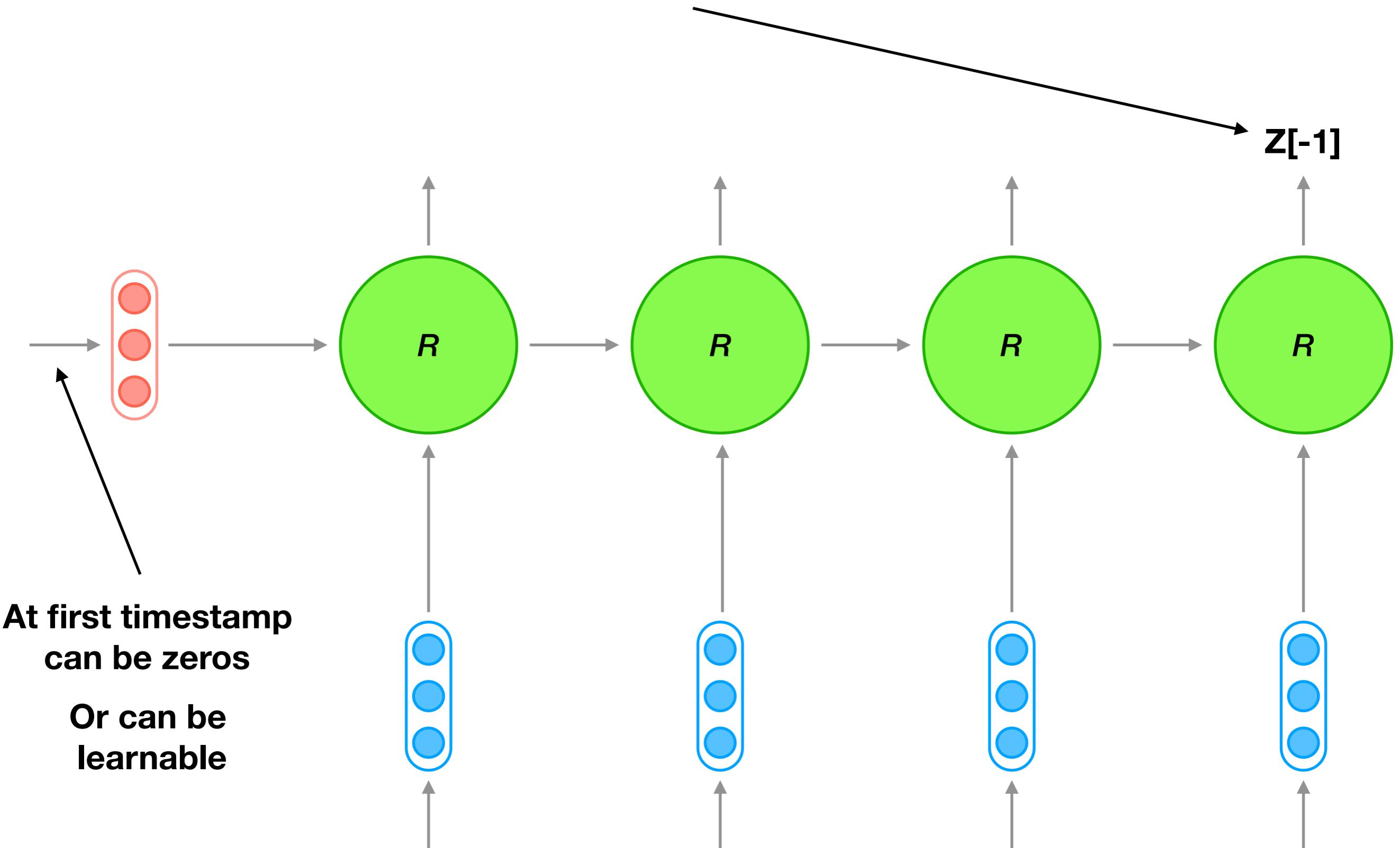


Reccurent Mechanism

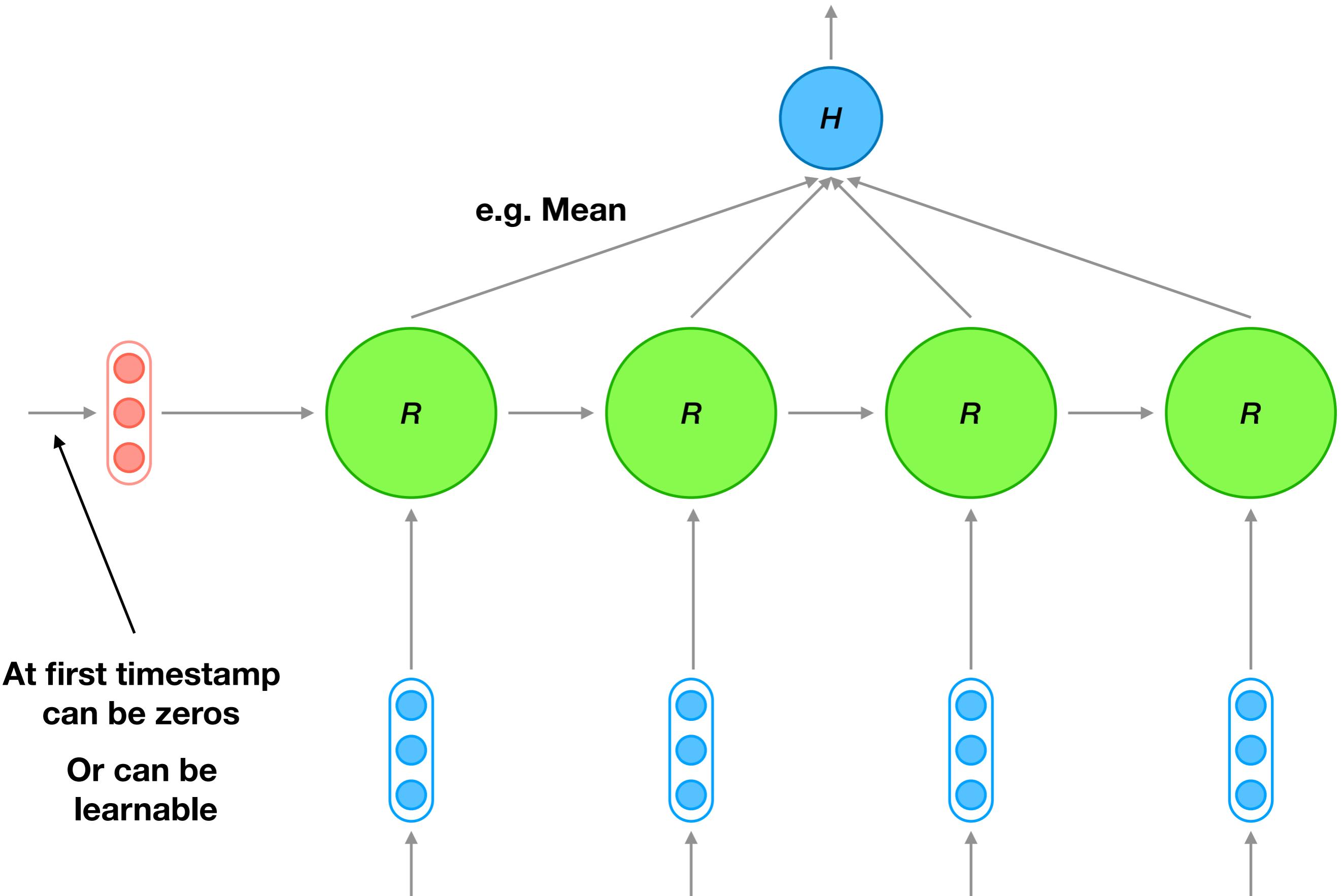


Reccurent Mechanism

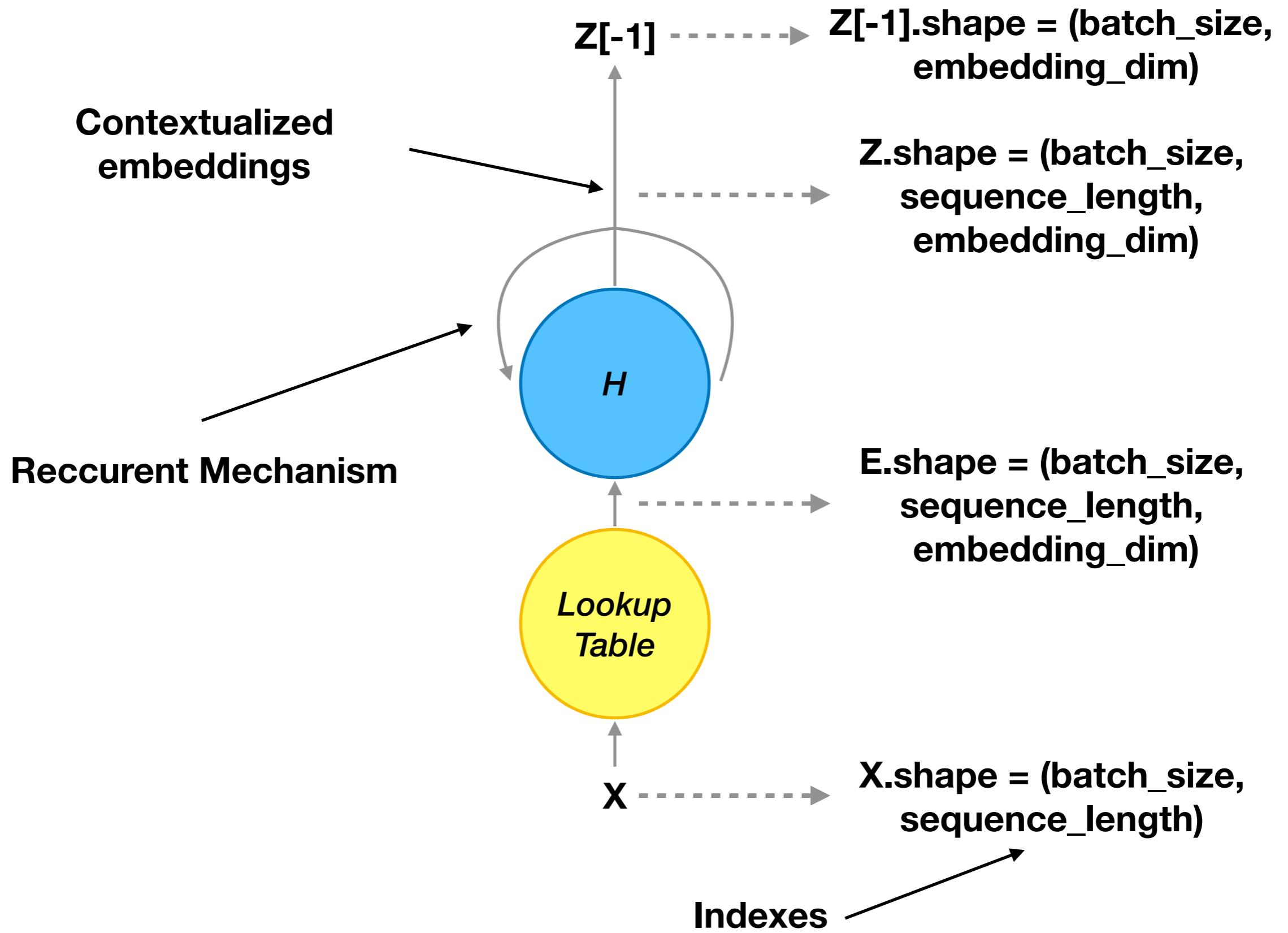
Now we can take last vector from sequence



Reccurent Mechanism

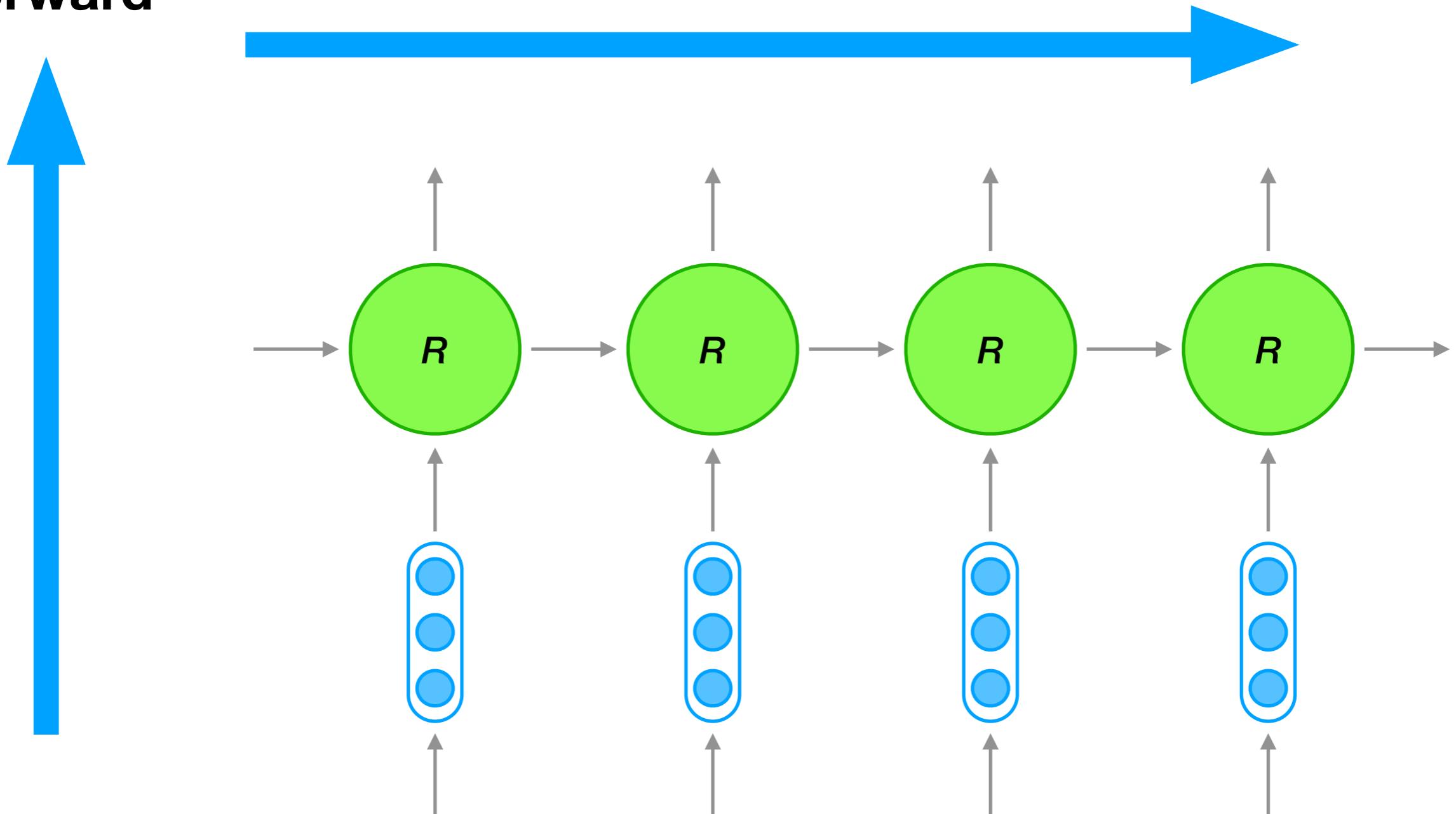


Recurrent Neural Network



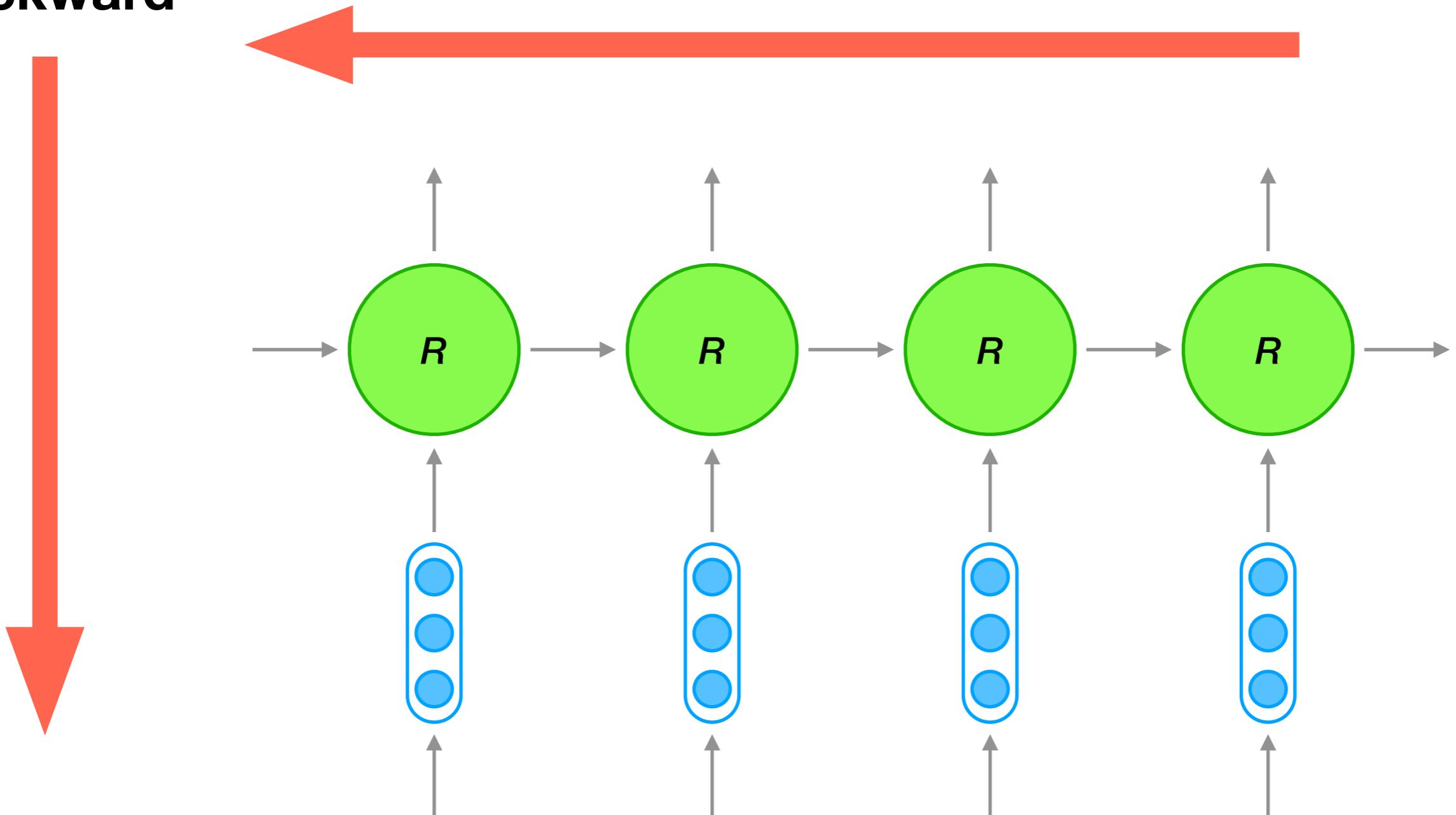
BPTT

Forward



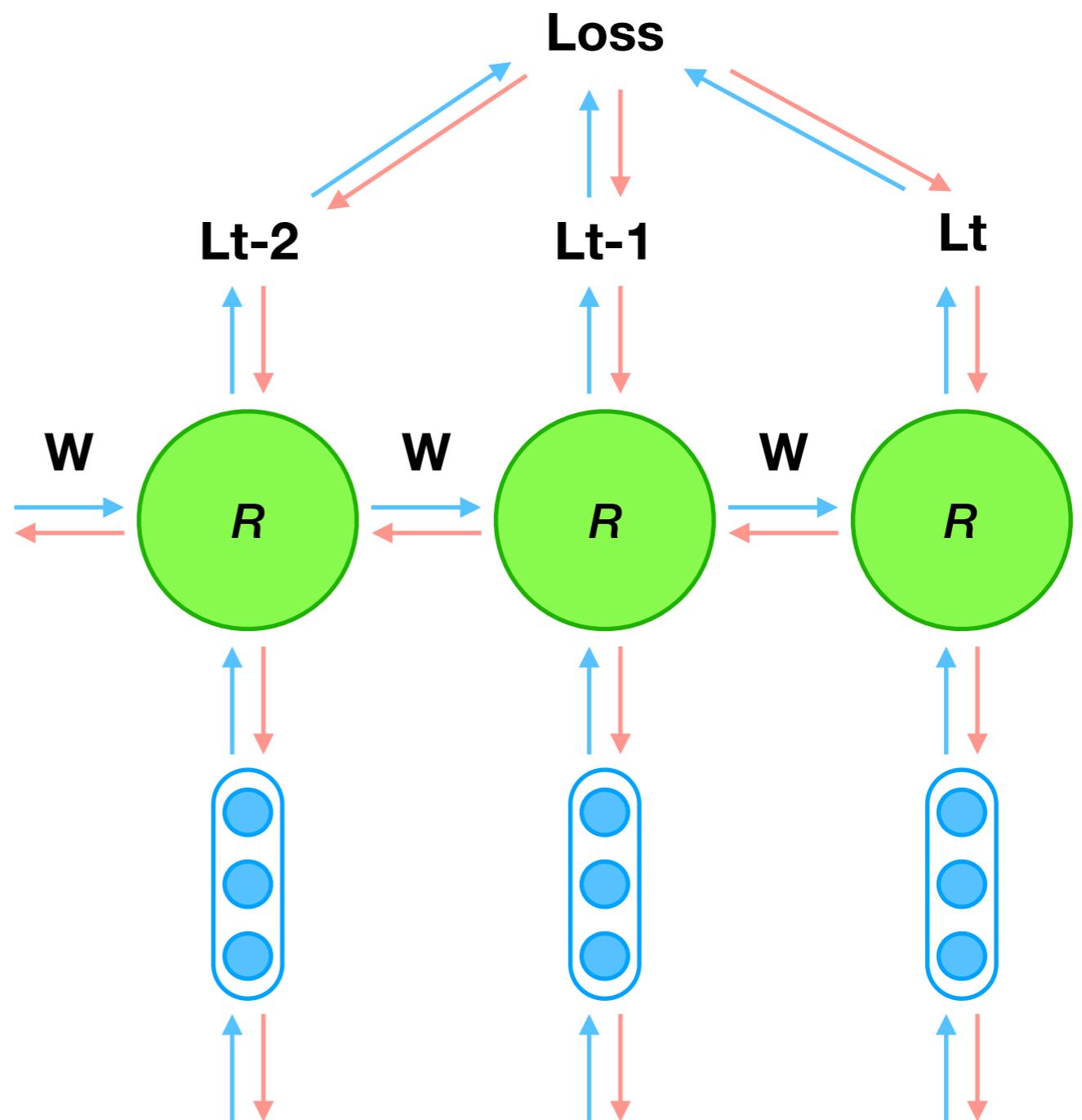
BPTT

Backward



BPTT

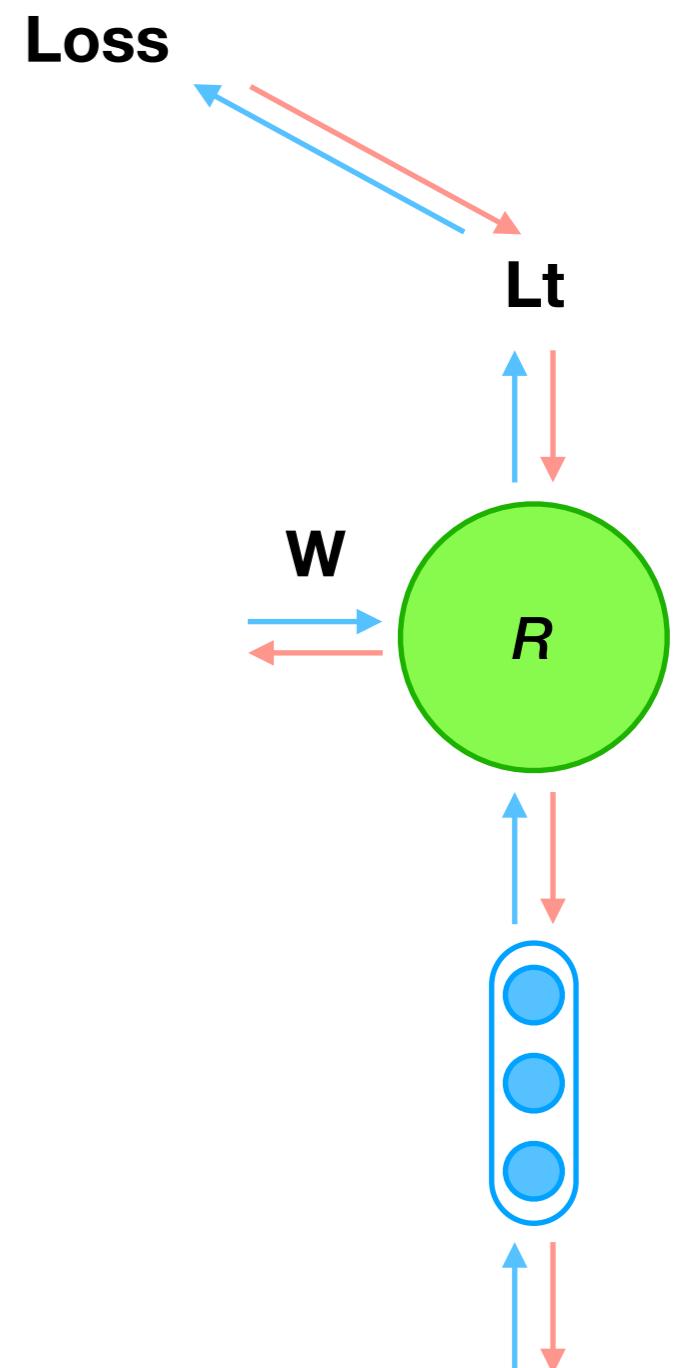
$$\frac{\partial L}{\partial W} = \sum_{i=0}^T \frac{\partial L_i}{\partial W}$$



BPTT

$$\frac{\partial L}{\partial W} = \sum_{i=0}^T \frac{\partial L_i}{\partial W}$$

$$\frac{\partial L_t}{\partial W} = \frac{\partial L_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial h_t} \frac{\partial h_t}{\partial W}$$



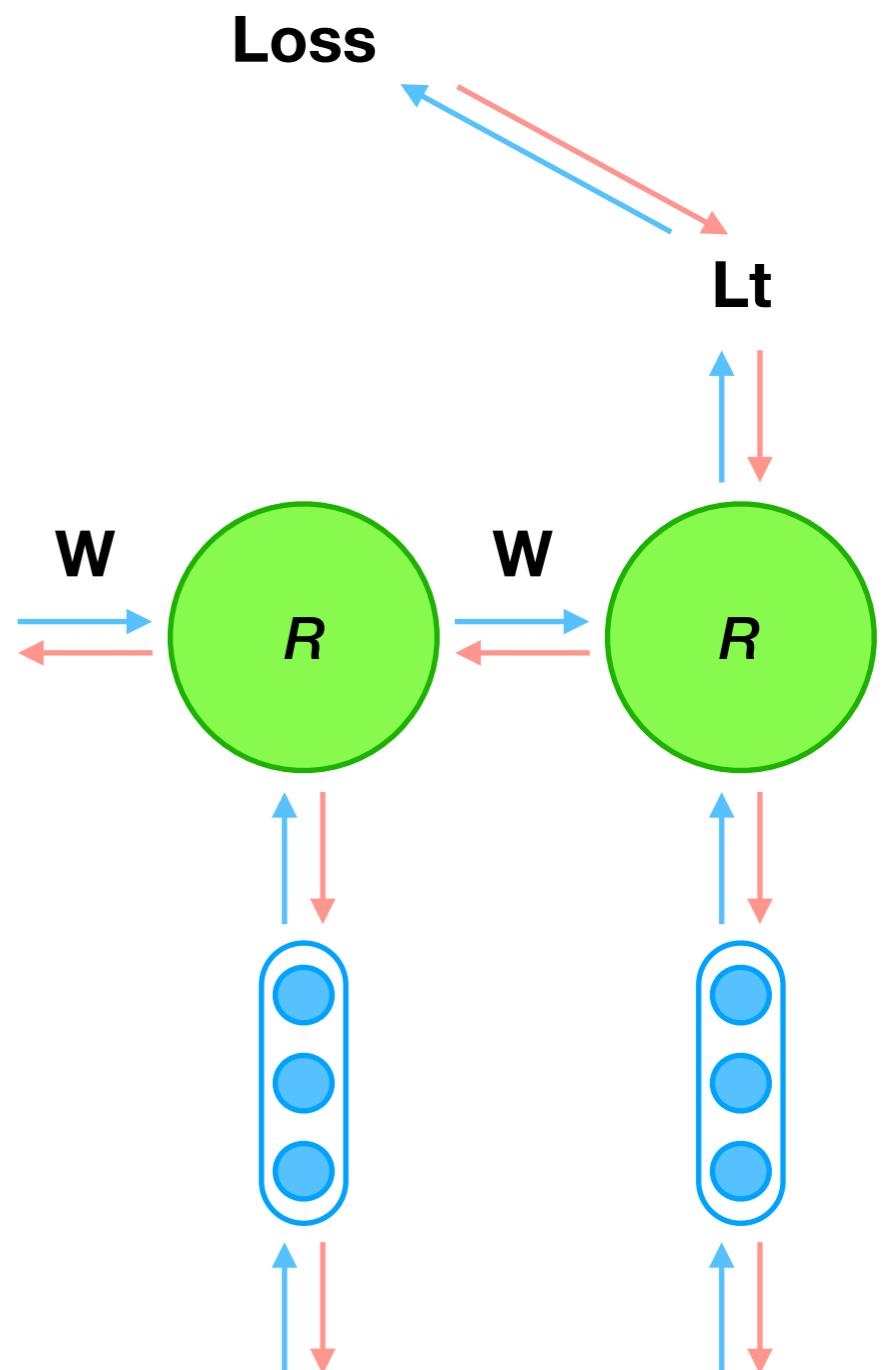
BPTT

$$\frac{\partial L}{\partial W} = \sum_{i=0}^T \frac{\partial L_i}{\partial W}$$

$$\frac{\partial L_t}{\partial W} = \frac{\partial L_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial h_t} \frac{\partial h_t}{\partial W}$$

$$h_t = f_h(Vx_t + Wh_{t-1} + b_h)$$

This is **NOT** the only dependence!



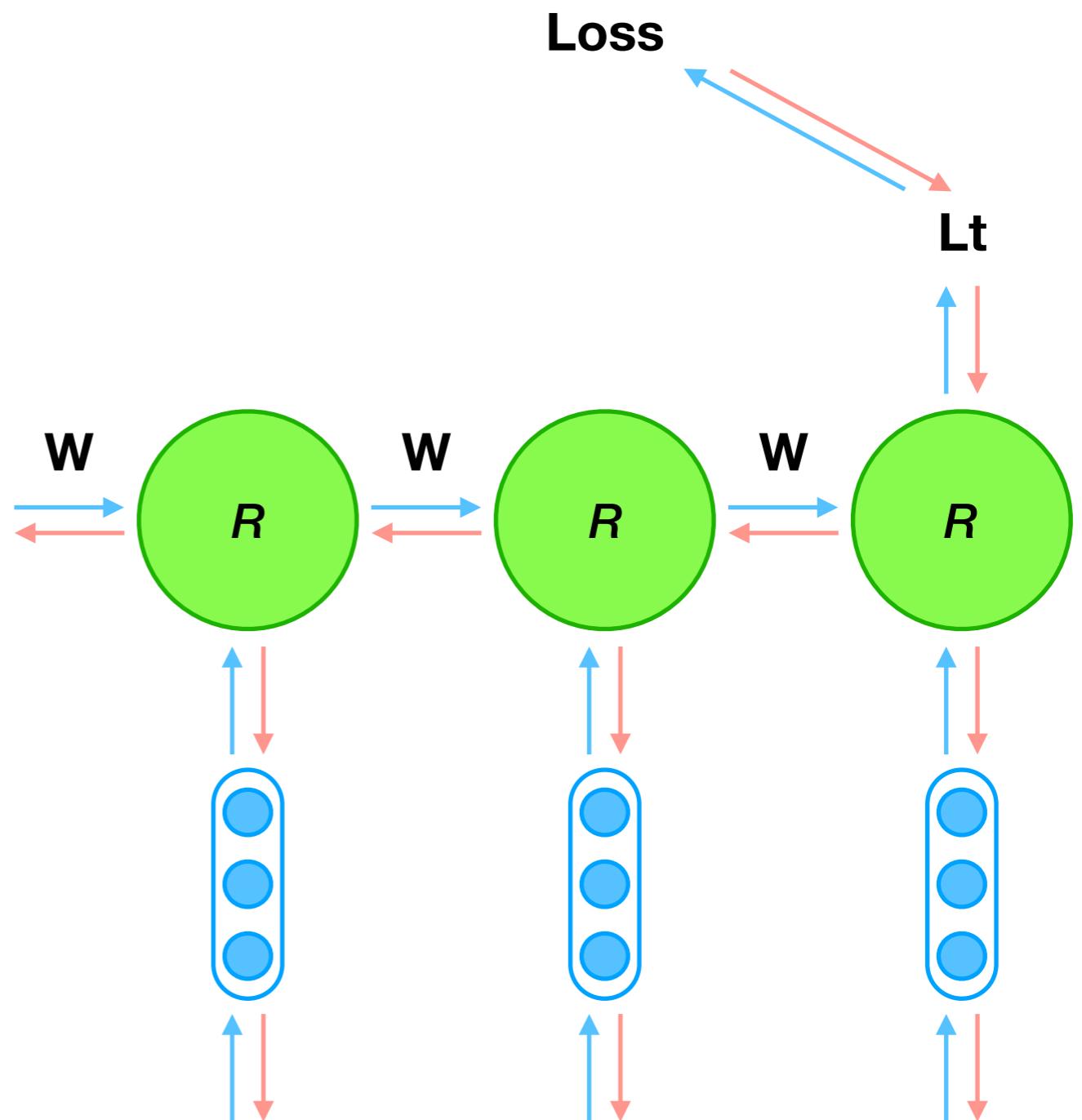
BPTT

$$\frac{\partial L}{\partial W} = \sum_{i=0}^T \frac{\partial L_i}{\partial W}$$

$$\frac{\partial L_t}{\partial W} = \frac{\partial L_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial h_t} \frac{\partial h_t}{\partial W}$$

$$h_t = f_h(Vx_t + Wh_{t-1} + b_h)$$

This is **NOT** the only dependence!



$$\frac{\partial L_t}{\partial W} = \frac{\partial L_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial h_t} \left(\frac{\partial h_t}{\partial W} + \frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial W} + \dots \right)$$

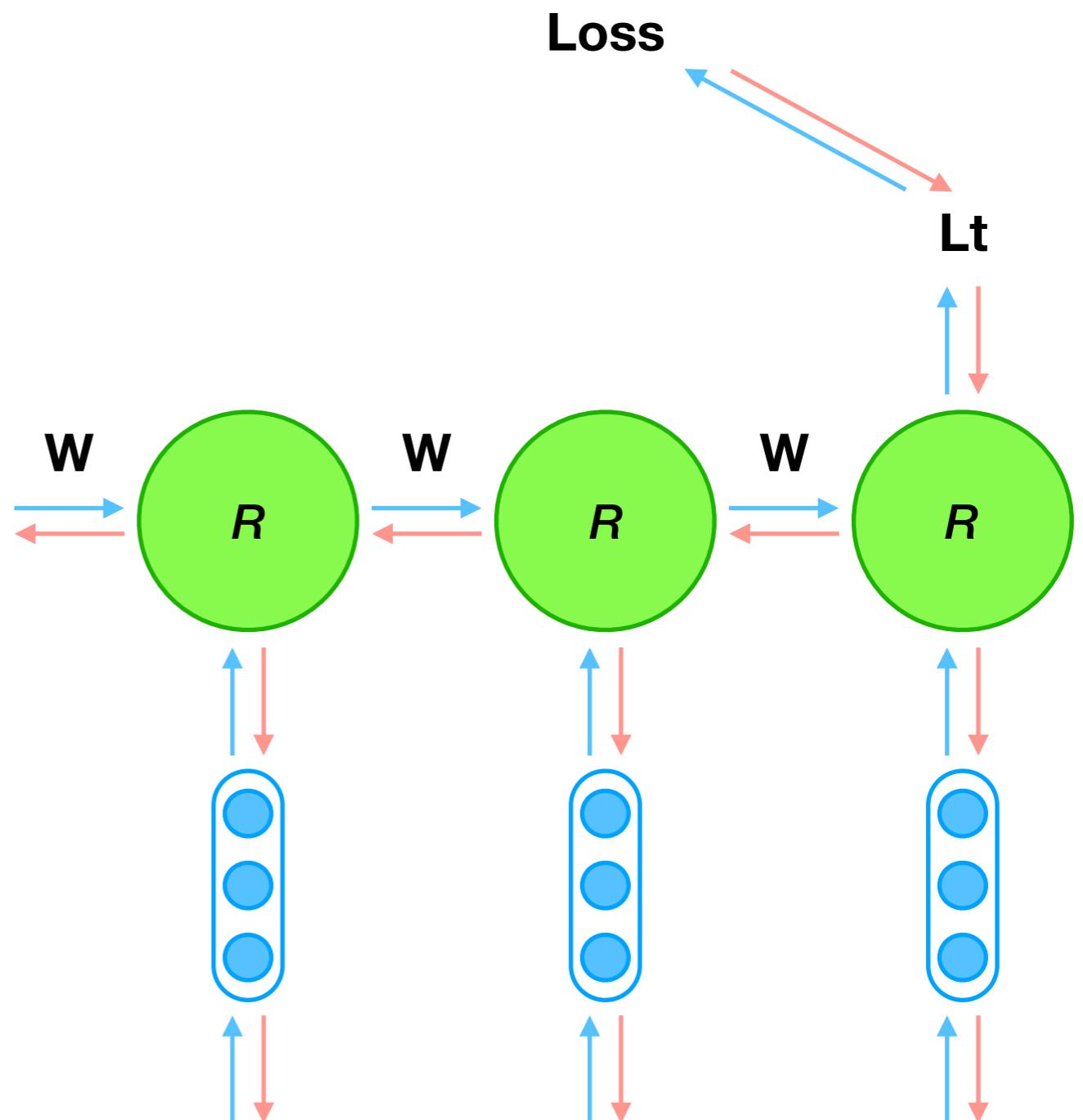
BPTT

$$\frac{\partial L}{\partial W} = \sum_{i=0}^T \frac{\partial L_i}{\partial W}$$

$$\frac{\partial L_t}{\partial W} = \frac{\partial L_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial h_t} \frac{\partial h_t}{\partial W}$$

$$h_t = f_h(Vx_t + Wh_{t-1} + b_h)$$

This is **NOT** the only dependence!



$$\frac{\partial L_t}{\partial W} = \frac{\partial L_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial h_t} \sum_{k=0}^t \left(\prod_{i=k+1}^t \frac{\partial h_i}{\partial h_{i-1}} \right) \frac{\partial h_k}{\partial W}$$

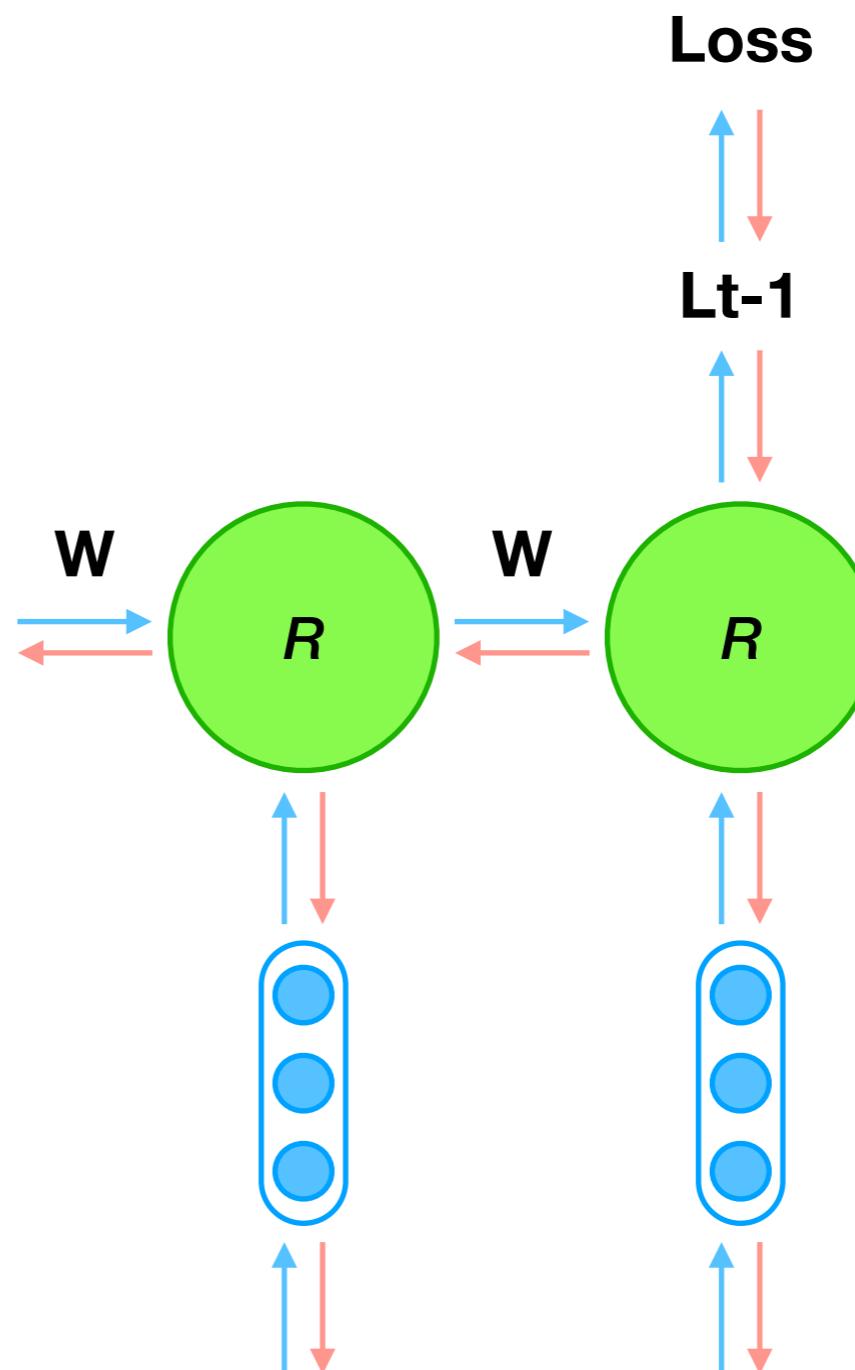
BPTT

$$\frac{\partial L}{\partial W} = \sum_{i=0}^T \frac{\partial L_i}{\partial W}$$

$$\frac{\partial L_t}{\partial W} = \frac{\partial L_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial h_t} \frac{\partial h_t}{\partial W}$$

$$h_t = f_h(Vx_t + Wh_{t-1} + b_h)$$

This is **NOT** the only dependence!



$$\frac{\partial L_t}{\partial W} = \frac{\partial L_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial h_t} \sum_{k=0}^t \left(\prod_{i=k+1}^t \frac{\partial h_i}{\partial h_{i-1}} \right) \frac{\partial h_k}{\partial W}$$

Gradients

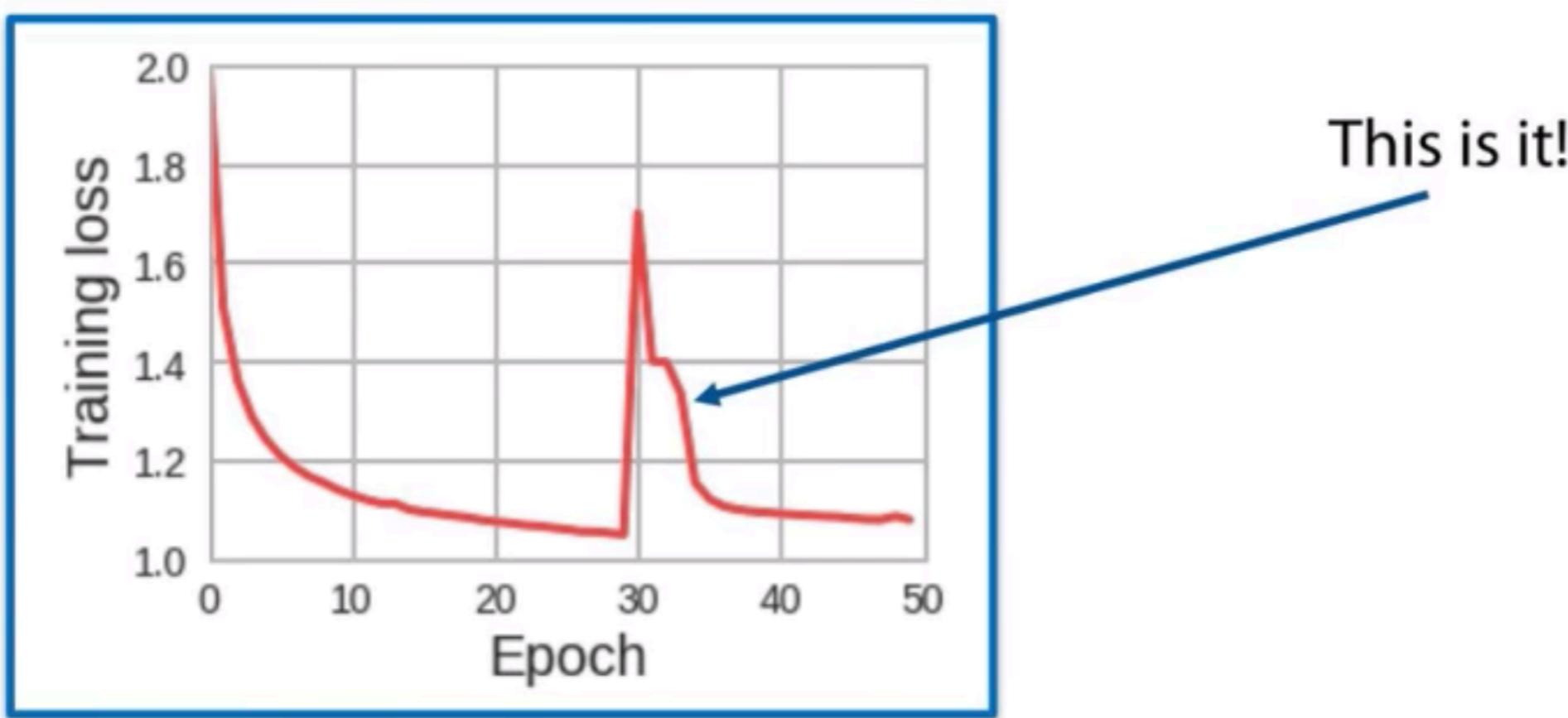
$$\frac{\partial L_t}{\partial W} \propto \sum_{k=0}^t \left(\prod_{i=k+1}^t \frac{\partial h_i}{\partial h_{i-1}} \right) \frac{\partial h_k}{\partial W}$$

$\left\| \frac{\partial h_i}{\partial h_{i-1}} \right\|_2 < 1 \longrightarrow \text{Vanishing gradients}$

$\left\| \frac{\partial h_i}{\partial h_{i-1}} \right\|_2 > 1 \longrightarrow \text{Exploding gradients}$

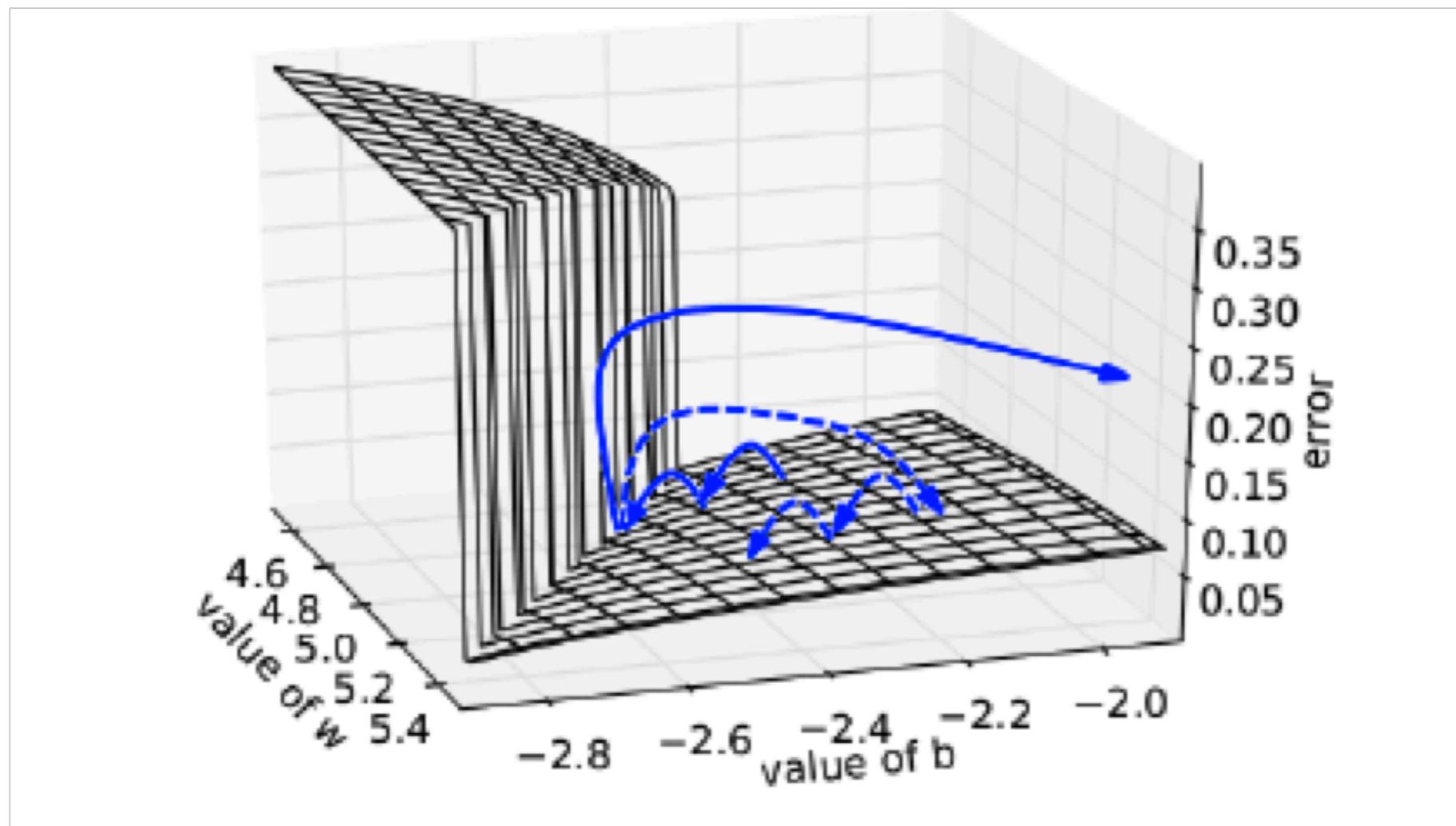
Exploding gradients

Unstable learning curve



If the gradients contain NaNs you end up
with NaNs in the weights

Exploding gradients



Gradient Clipping

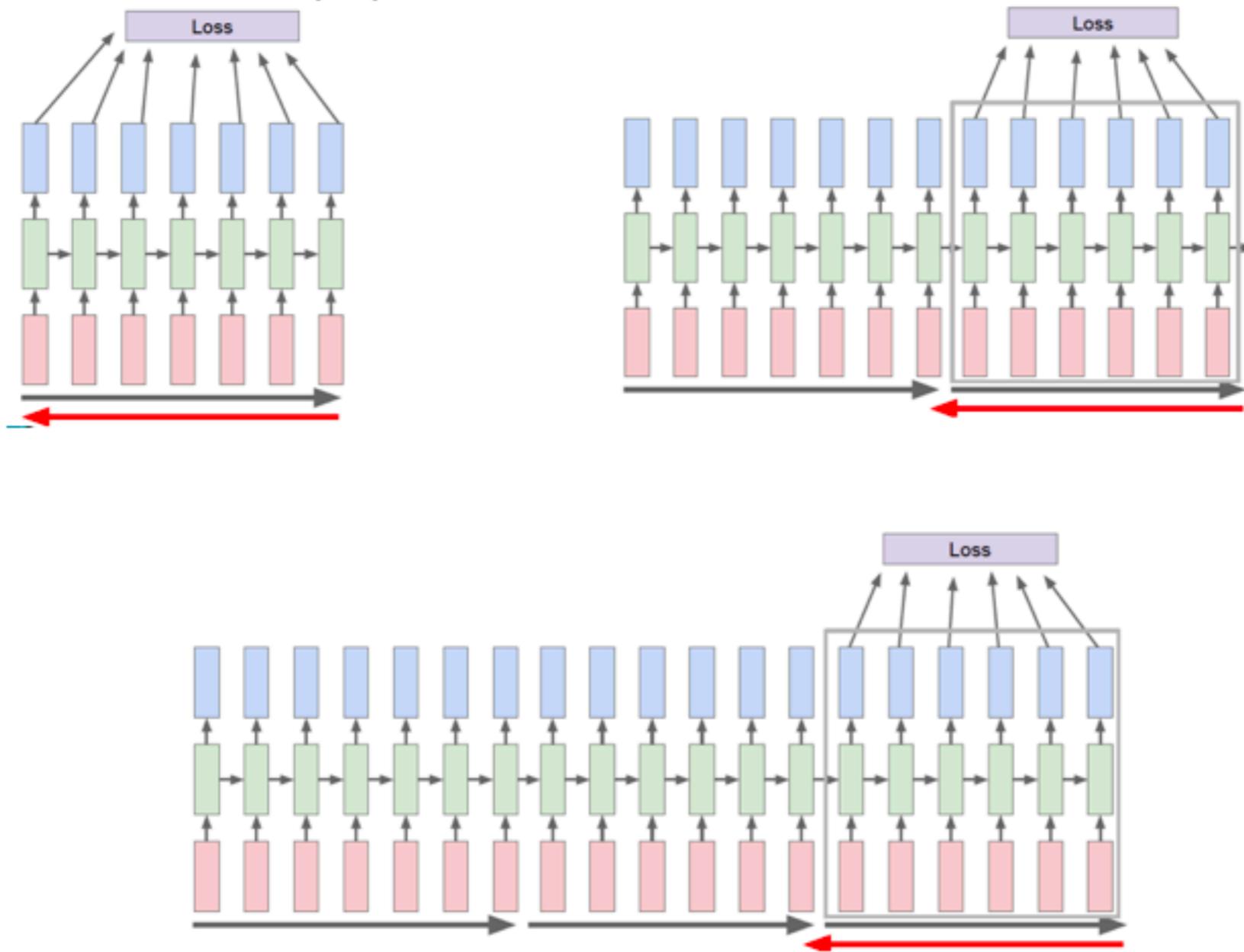
Gradient $g = \frac{\partial L}{\partial \theta}$, θ - all the network parameters

If $\|g\| > \text{threshold}$:

$$g \leftarrow \frac{\text{threshold}}{\|g\|} g$$

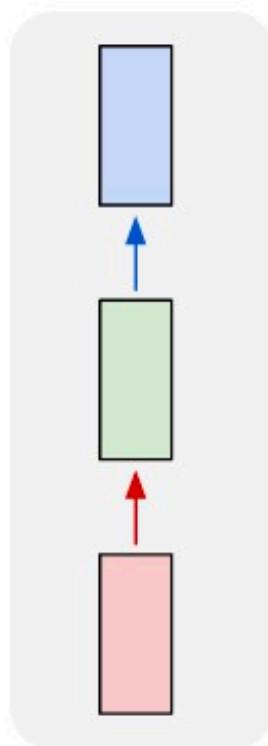
Simple but still very effective!

Truncated BPTT

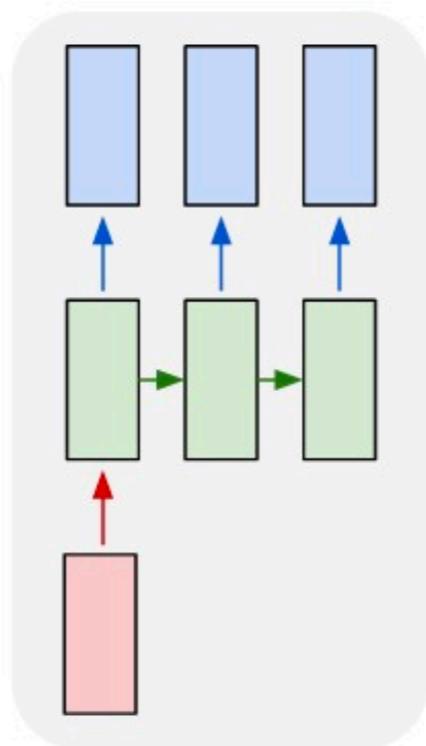


Recurrent Neural Network

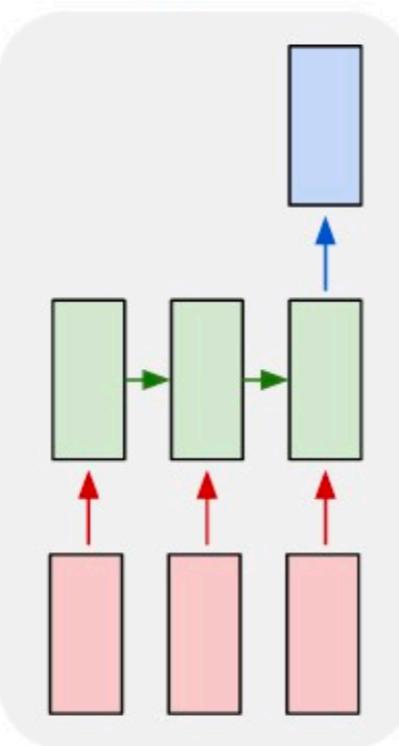
one to one



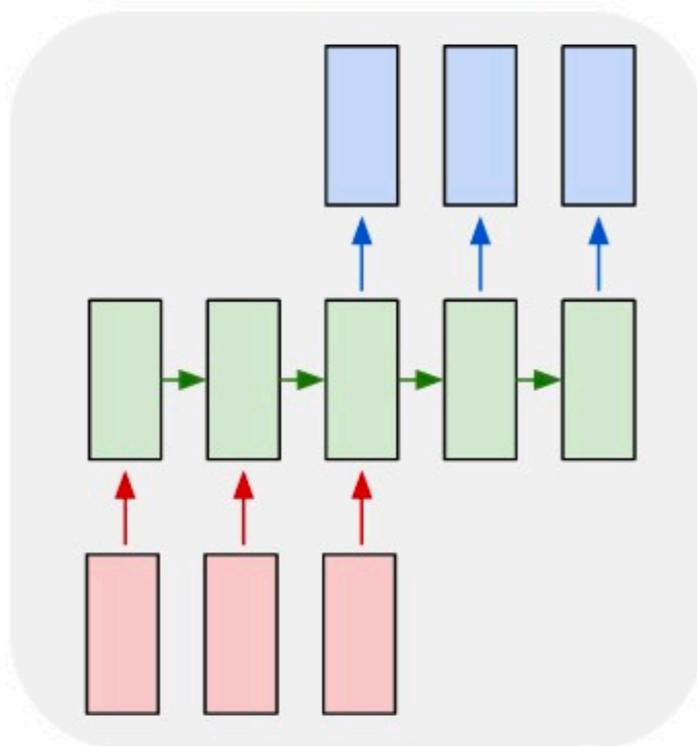
one to many



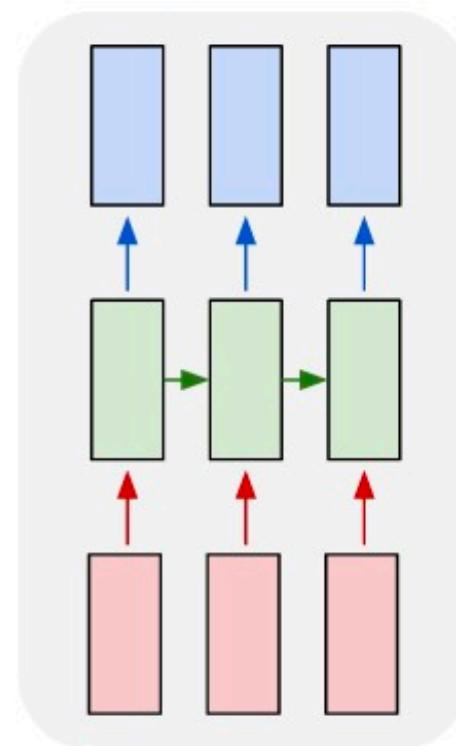
many to one



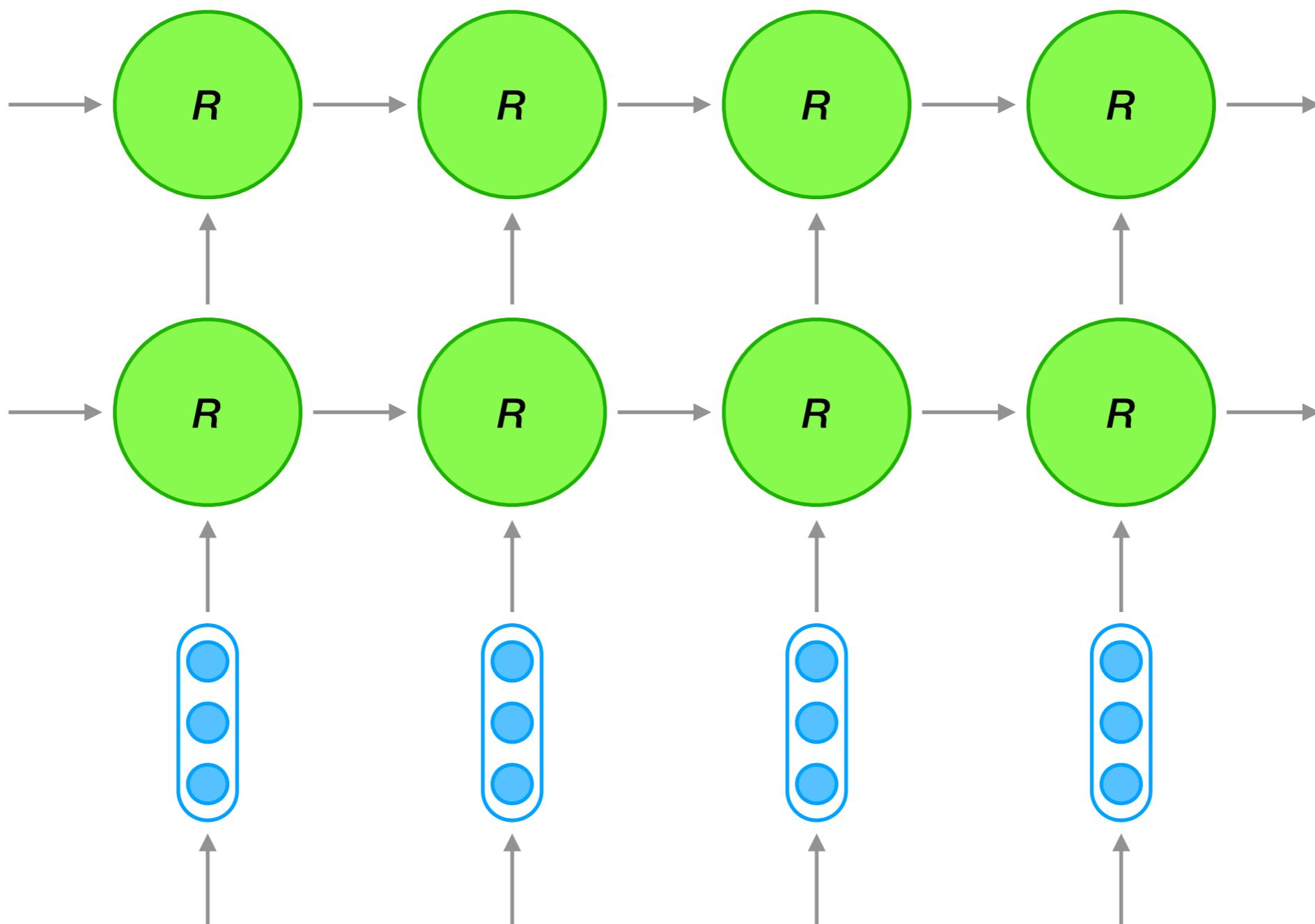
many to many



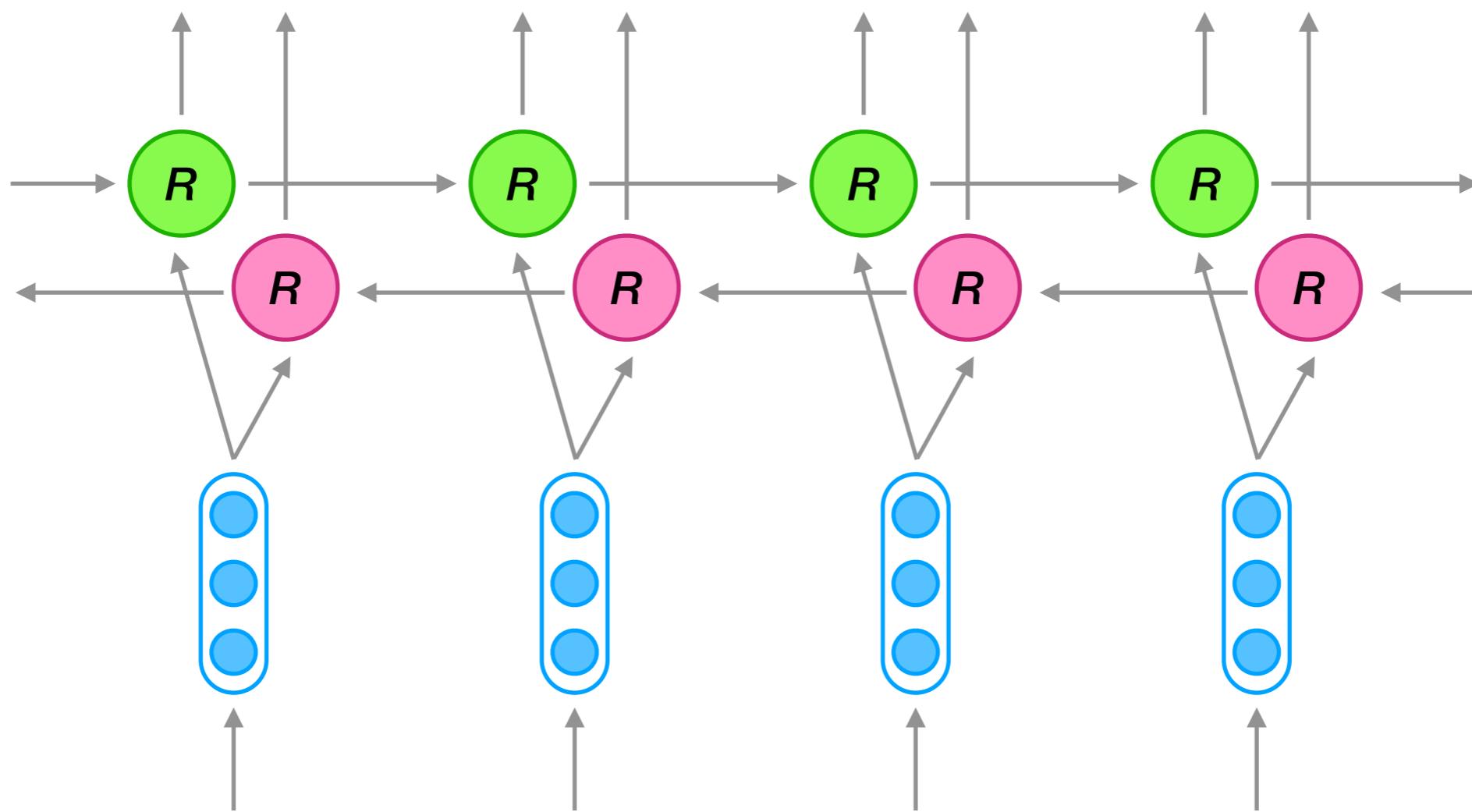
many to many



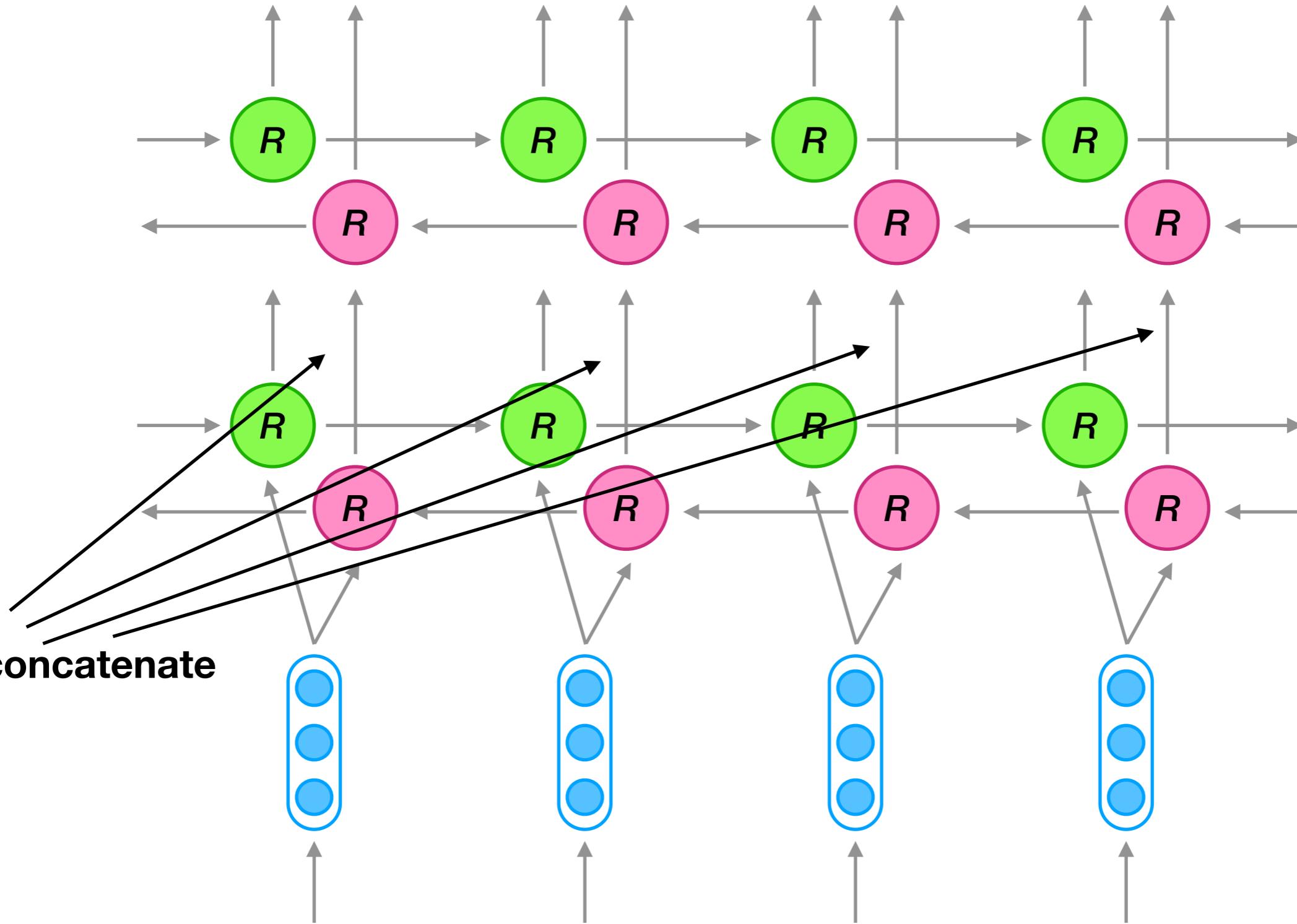
Deeper



Bidirectional



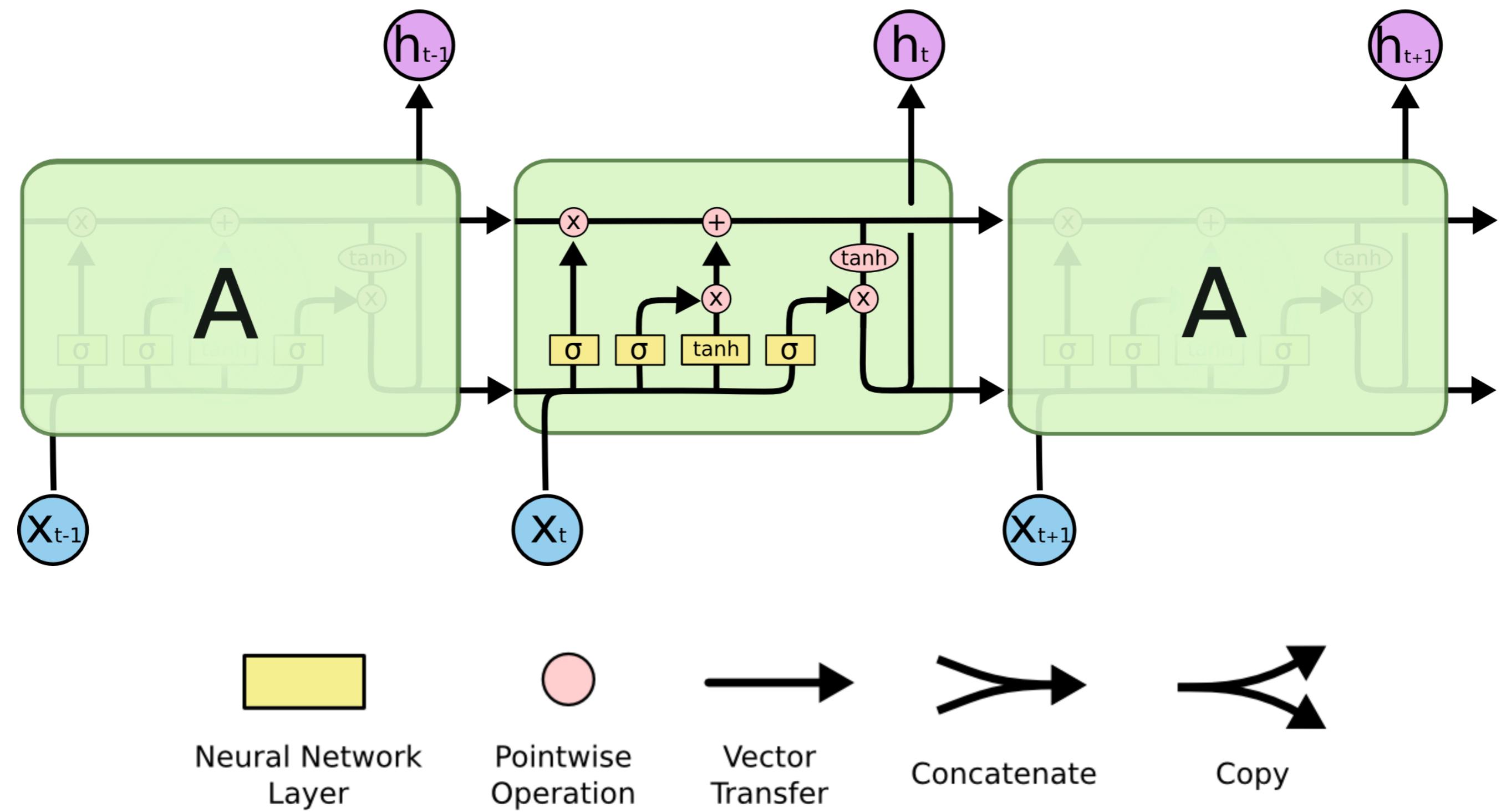
Deeper Bidirectional



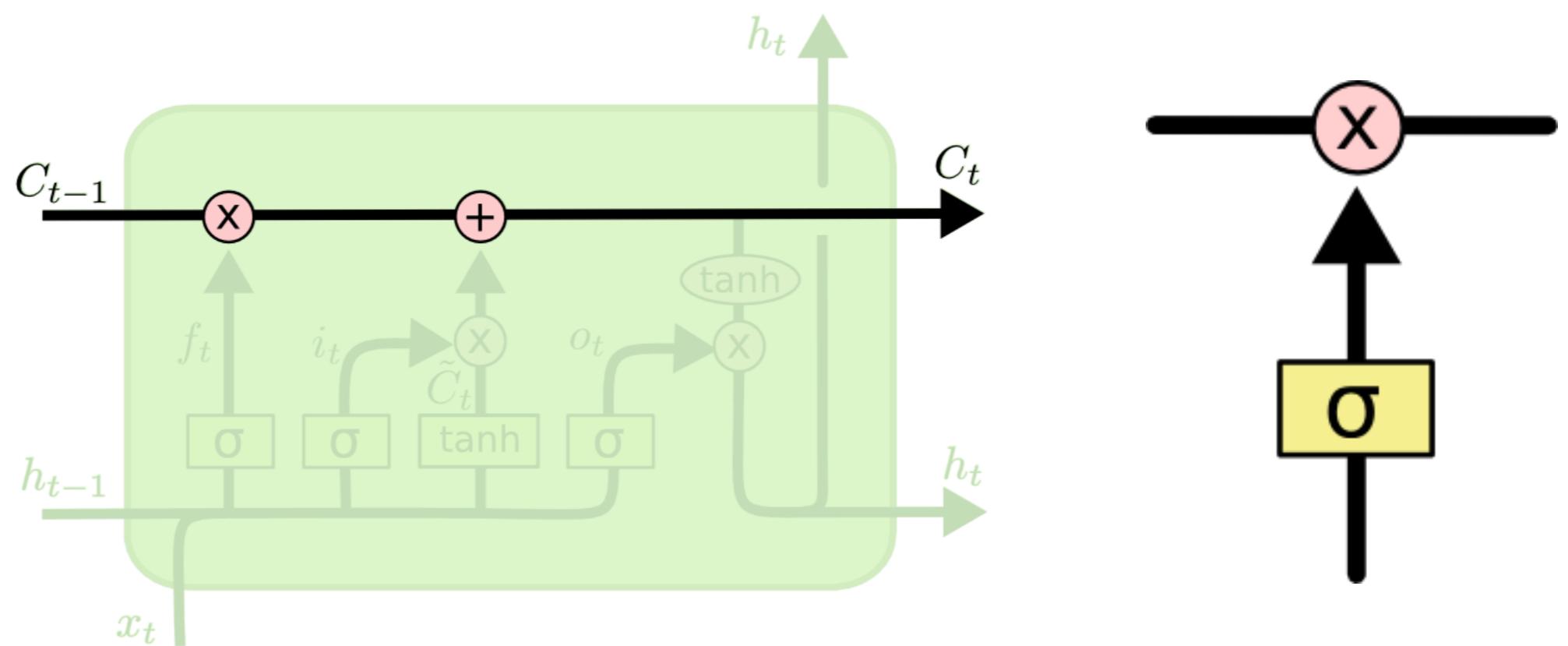
Bidirectional

Source	<START>	The	poor	don't	have	any	money	<END>
Forward	<START>	The	poor	don't	have	any	money	<END>
Backward	<END>	money	any	have	don't	poor	The	<START>

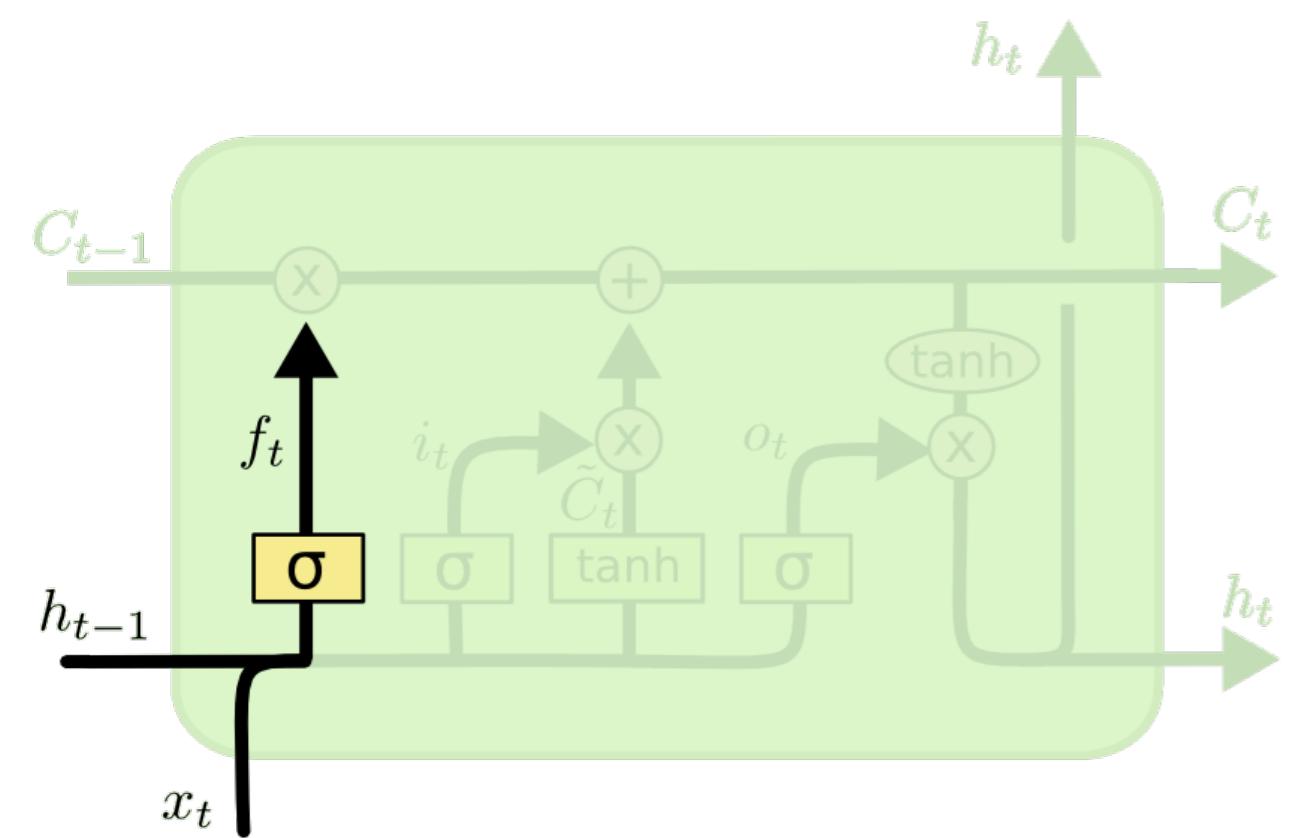
Long Short Term Memories



Long Short Term Memories

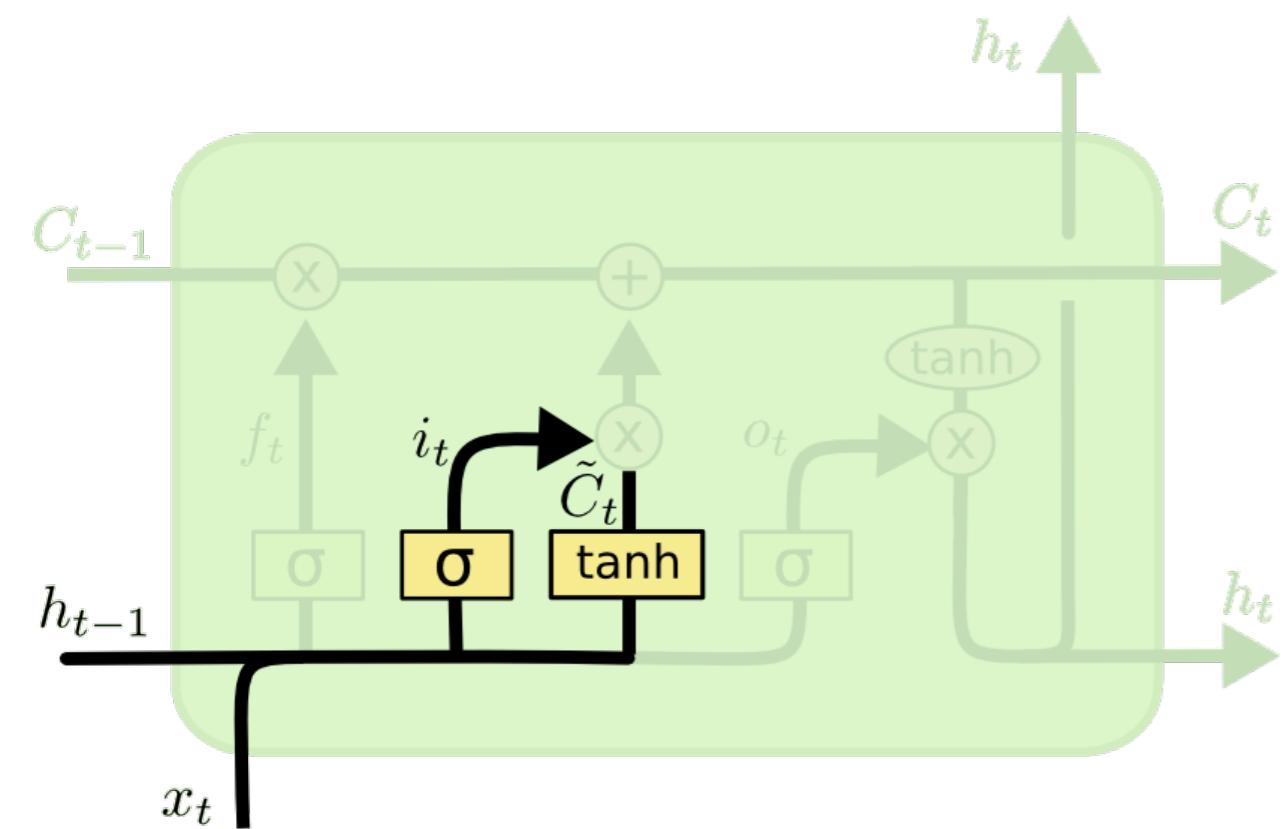


Long Short Term Memories



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

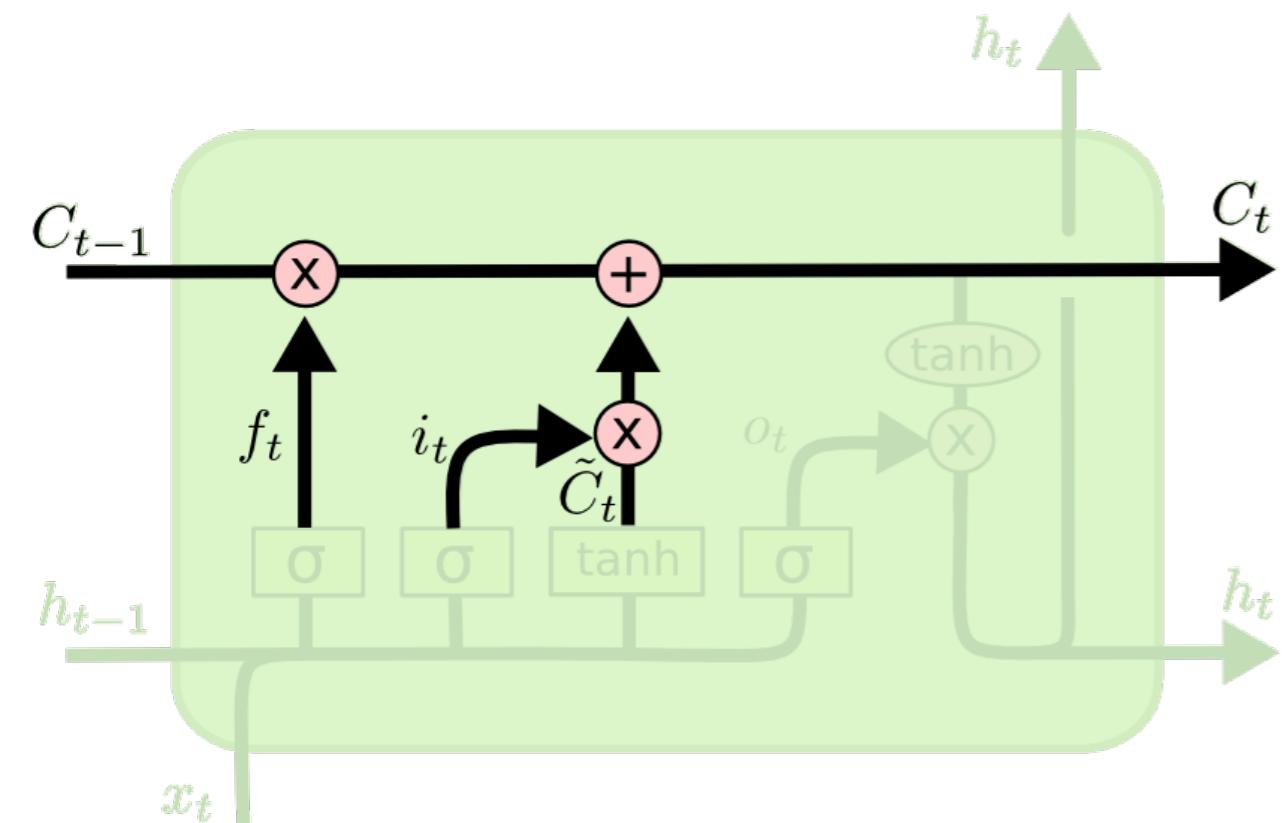
Long Short Term Memories



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

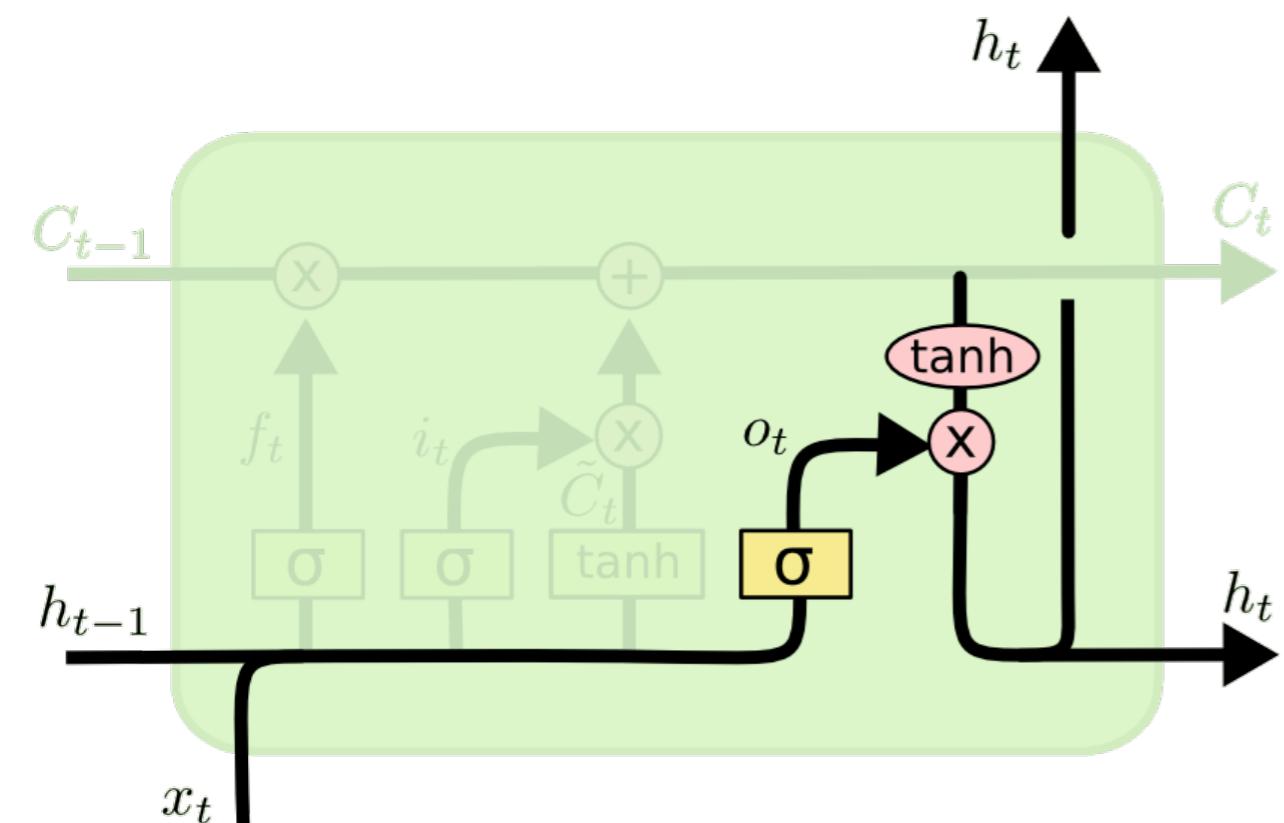
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

Long Short Term Memories



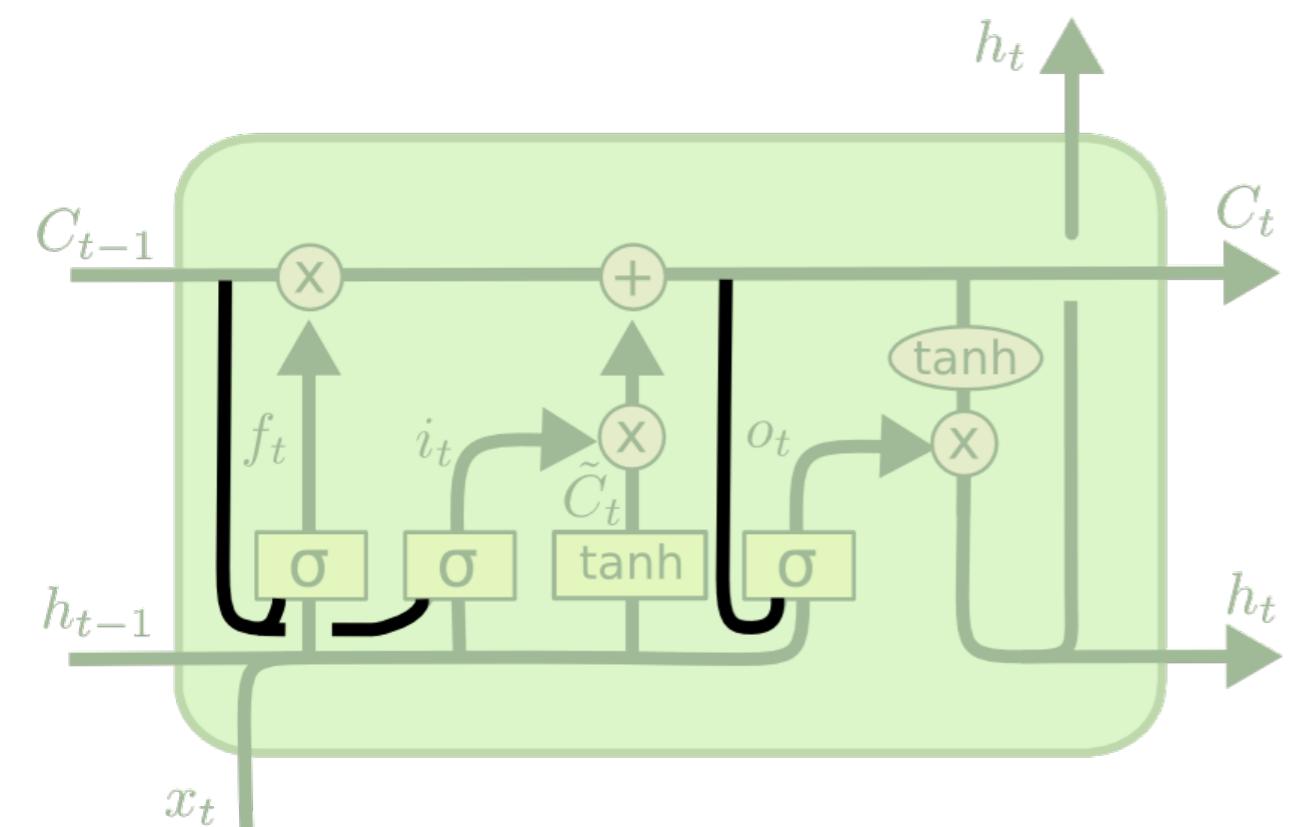
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Long Short Term Memories



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

Long Short Term Memories

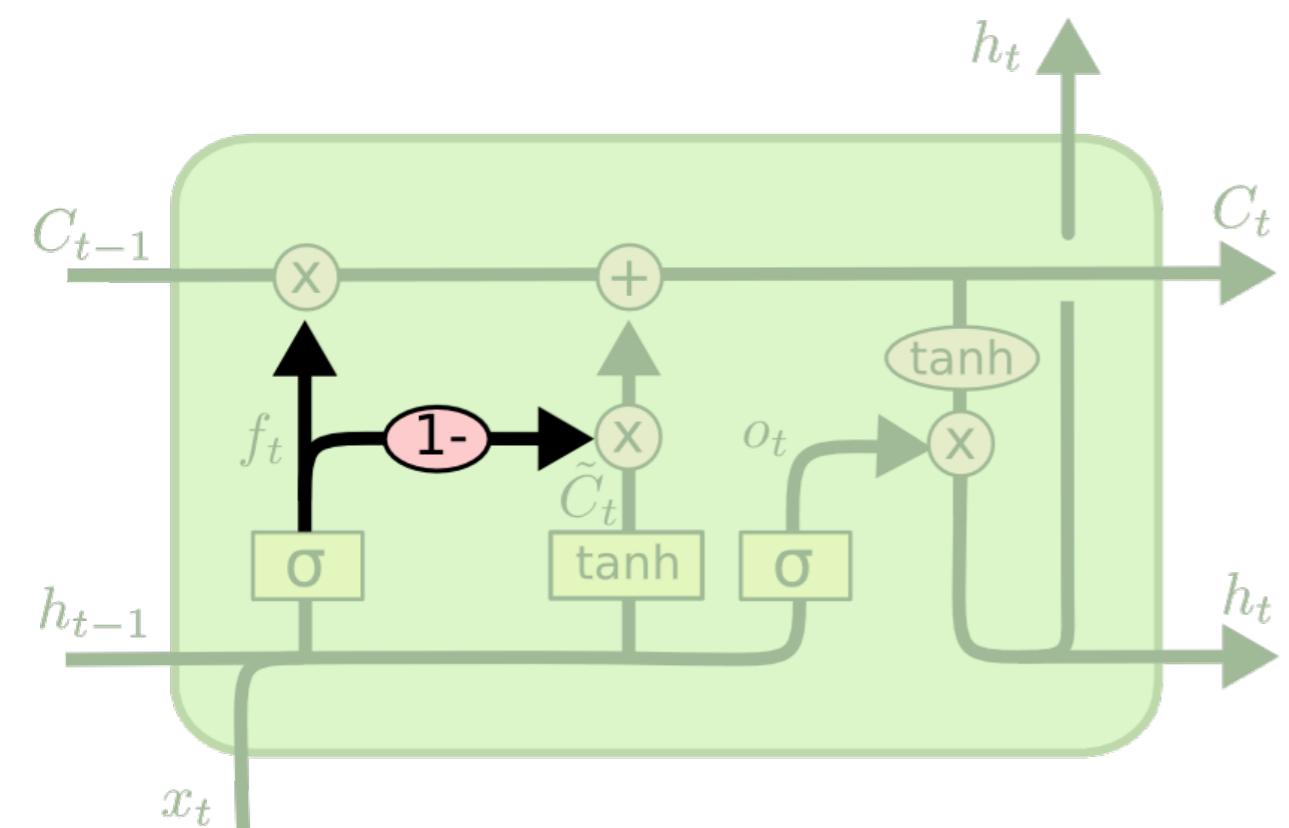


$$f_t = \sigma(W_f \cdot [C_{t-1}, h_{t-1}, x_t] + b_f)$$

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

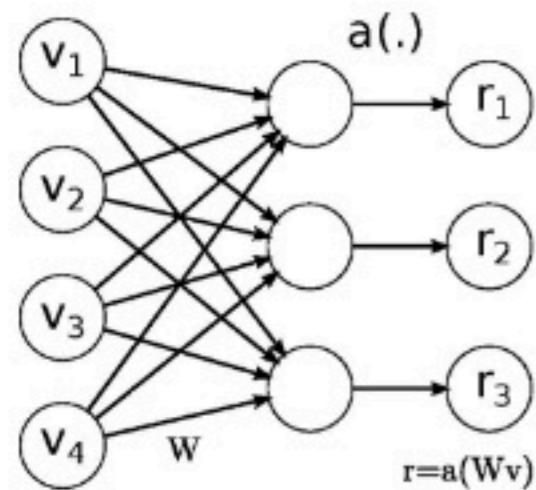
$$o_t = \sigma(W_o \cdot [C_t, h_{t-1}, x_t] + b_o)$$

Long Short Term Memories

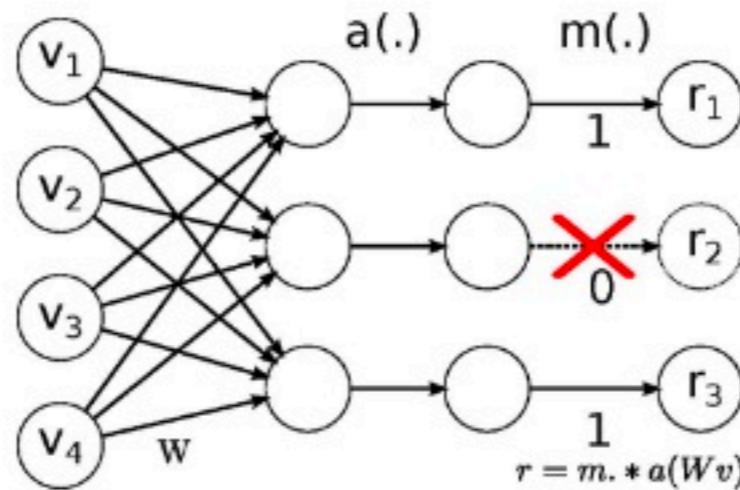


$$C_t = f_t * C_{t-1} + (1 - f_t) * \tilde{C}_t$$

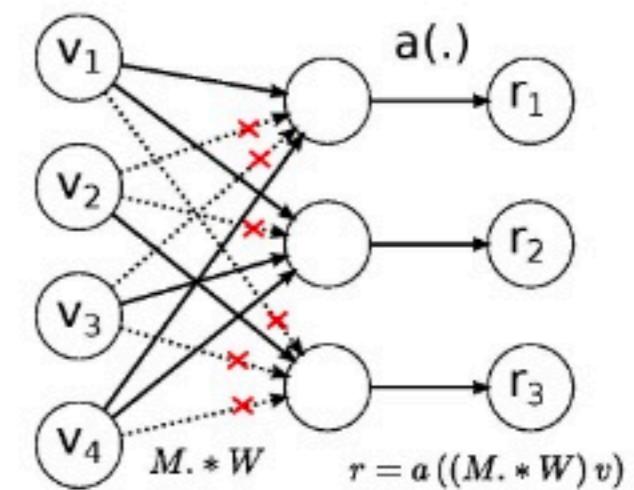
Dropout in RNN



No-Drop Network



DropOut Network



DropConnect Network

Layer Normalization

Batch Normalization



Layer Normalization

$$\mu_j = \frac{1}{m} \sum_{i=1}^m x_{ij}$$
$$\sigma_j^2 = \frac{1}{m} \sum_{i=1}^m (x_{ij} - \mu_j)^2$$
$$\hat{x}_{ij} = \frac{x_{ij} - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}}$$

$$\mu_i = \frac{1}{m} \sum_{j=1}^m x_{ij}$$
$$\sigma_i^2 = \frac{1}{m} \sum_{j=1}^m (x_{ij} - \mu_i)^2$$
$$\hat{x}_{ij} = \frac{x_{ij} - \mu_i}{\sqrt{\sigma_i^2 + \epsilon}}$$

