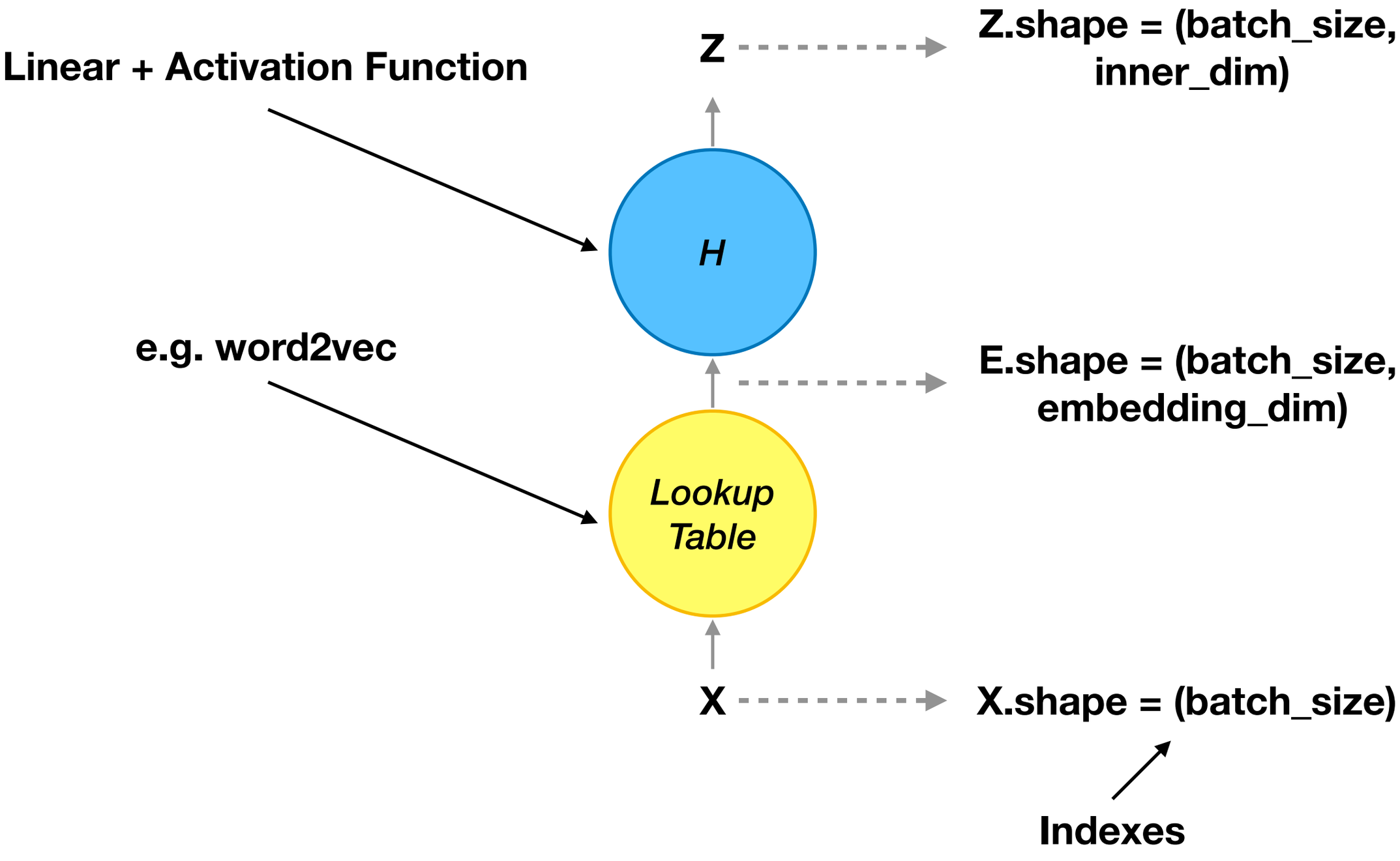
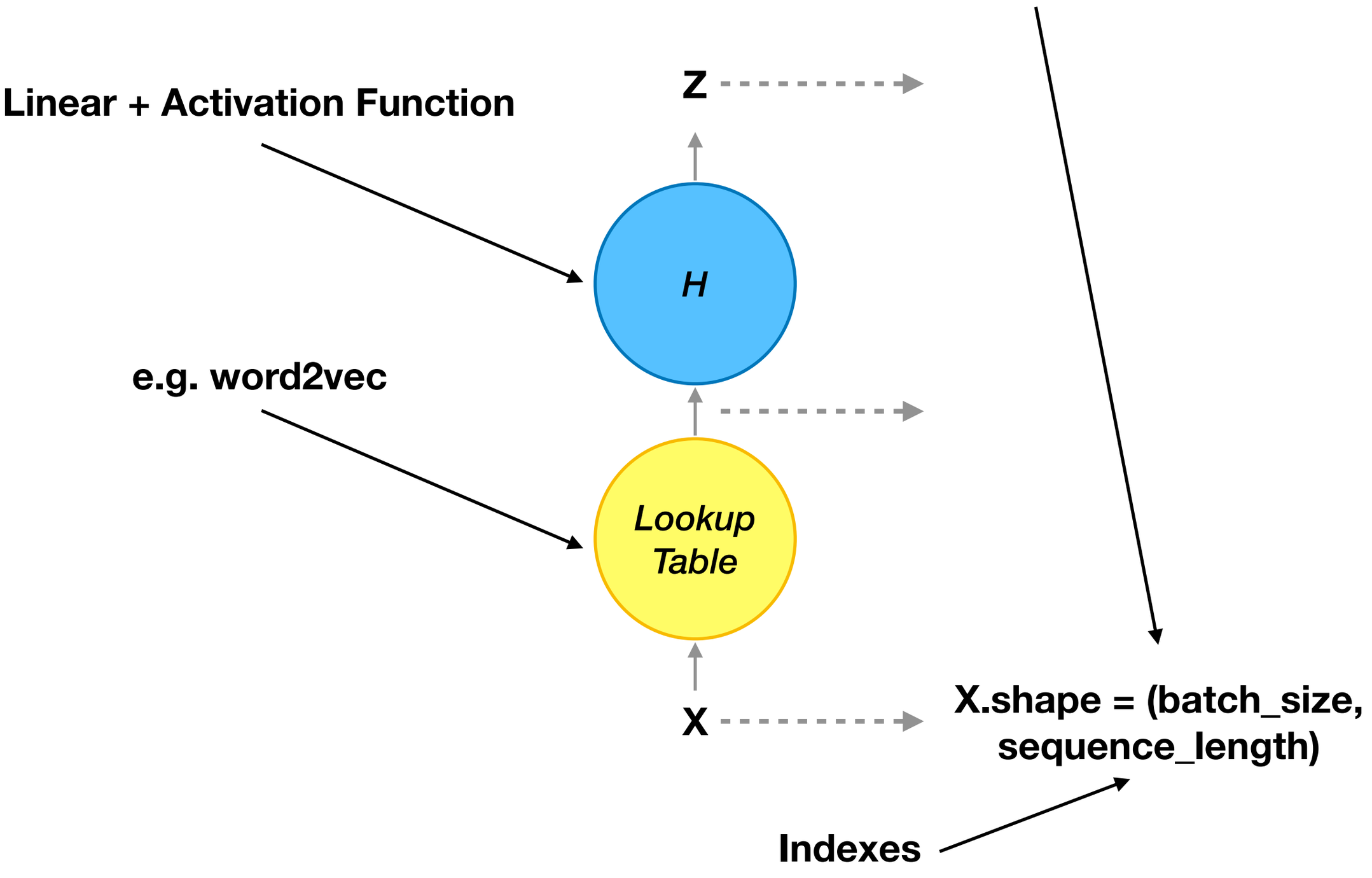
Recurrent Neural Network

Neural Network



Neural Network

What if we have different lengths?



Padding

[Мама, мыла, раму]

[Покупать, новый, телефон, затратно]

[Вау]

[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]

[Вау, PAD, 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]

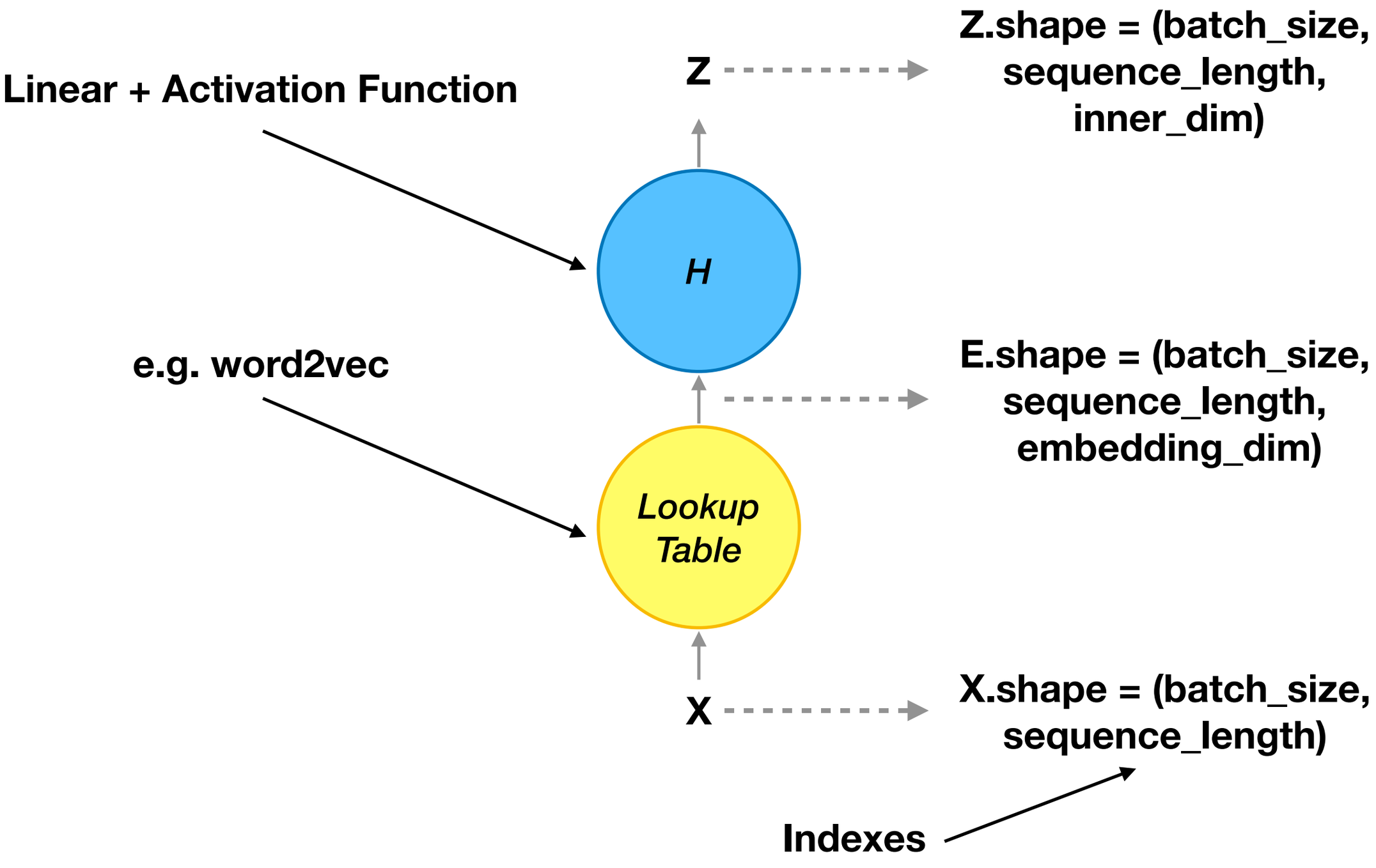
[Вау, PAD, PAD, PAD, PAD, PAD, PAD, PAD]

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

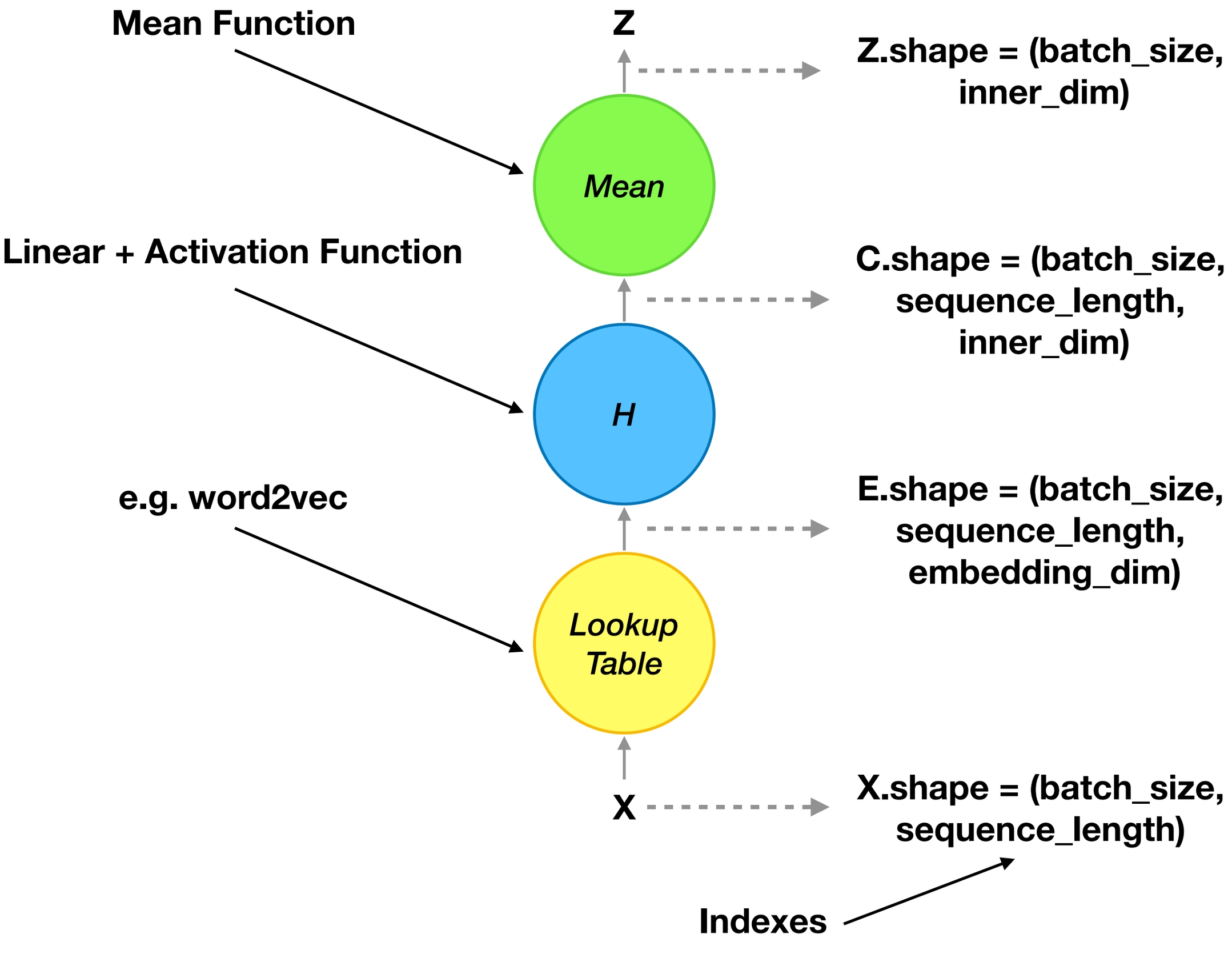
Indexing | **Collect word2index vocabulary**

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

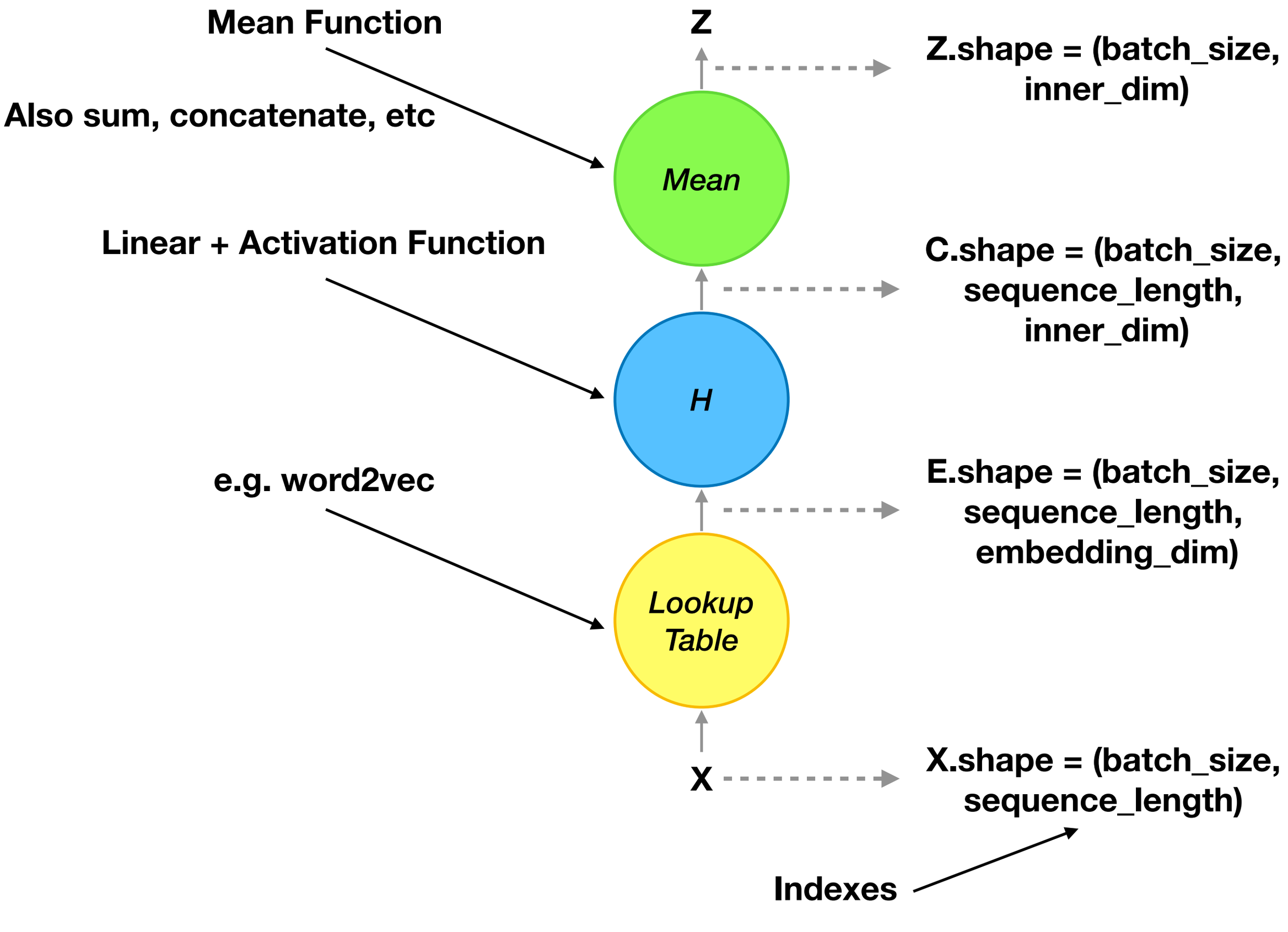
Neural Network



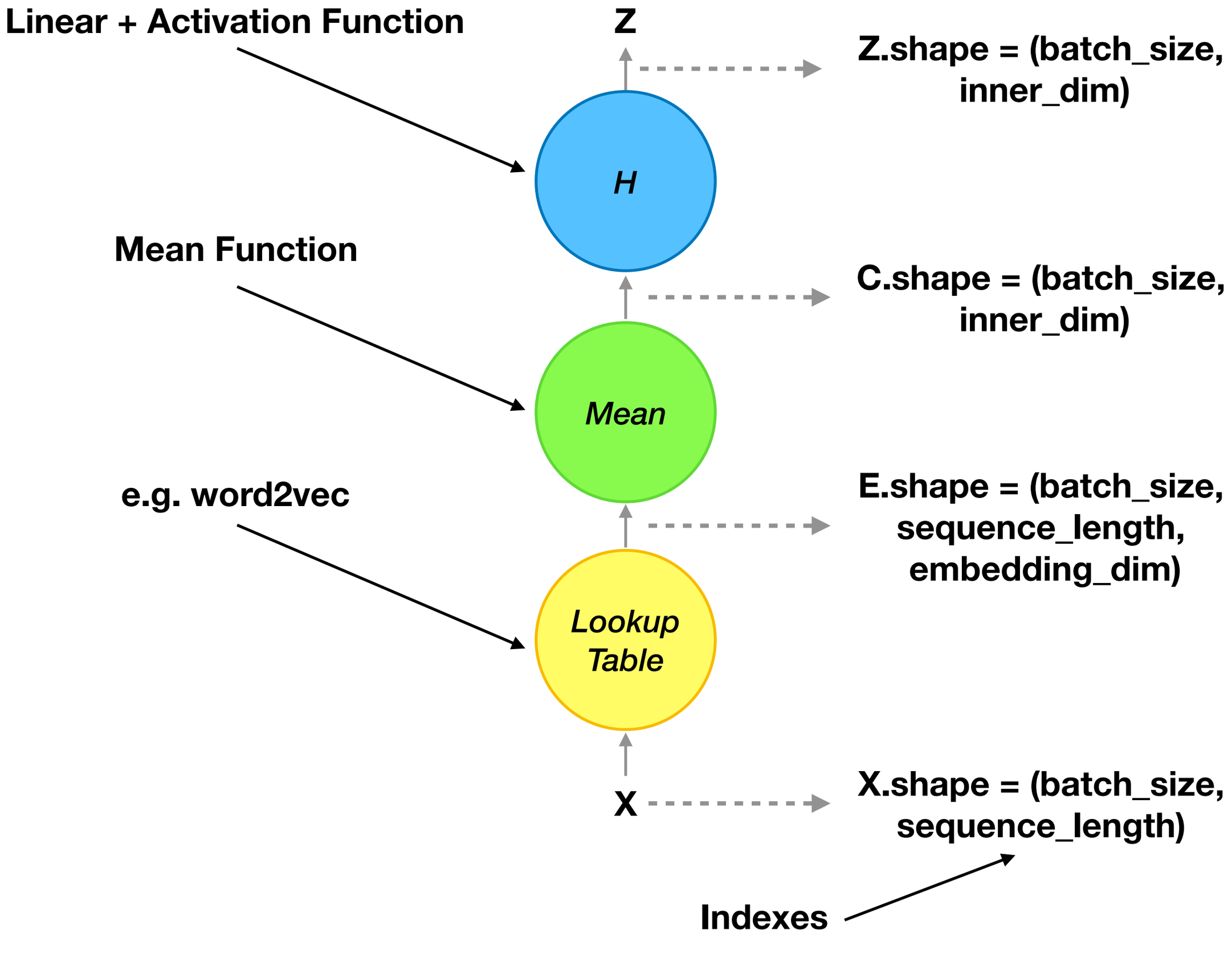
Neural Network



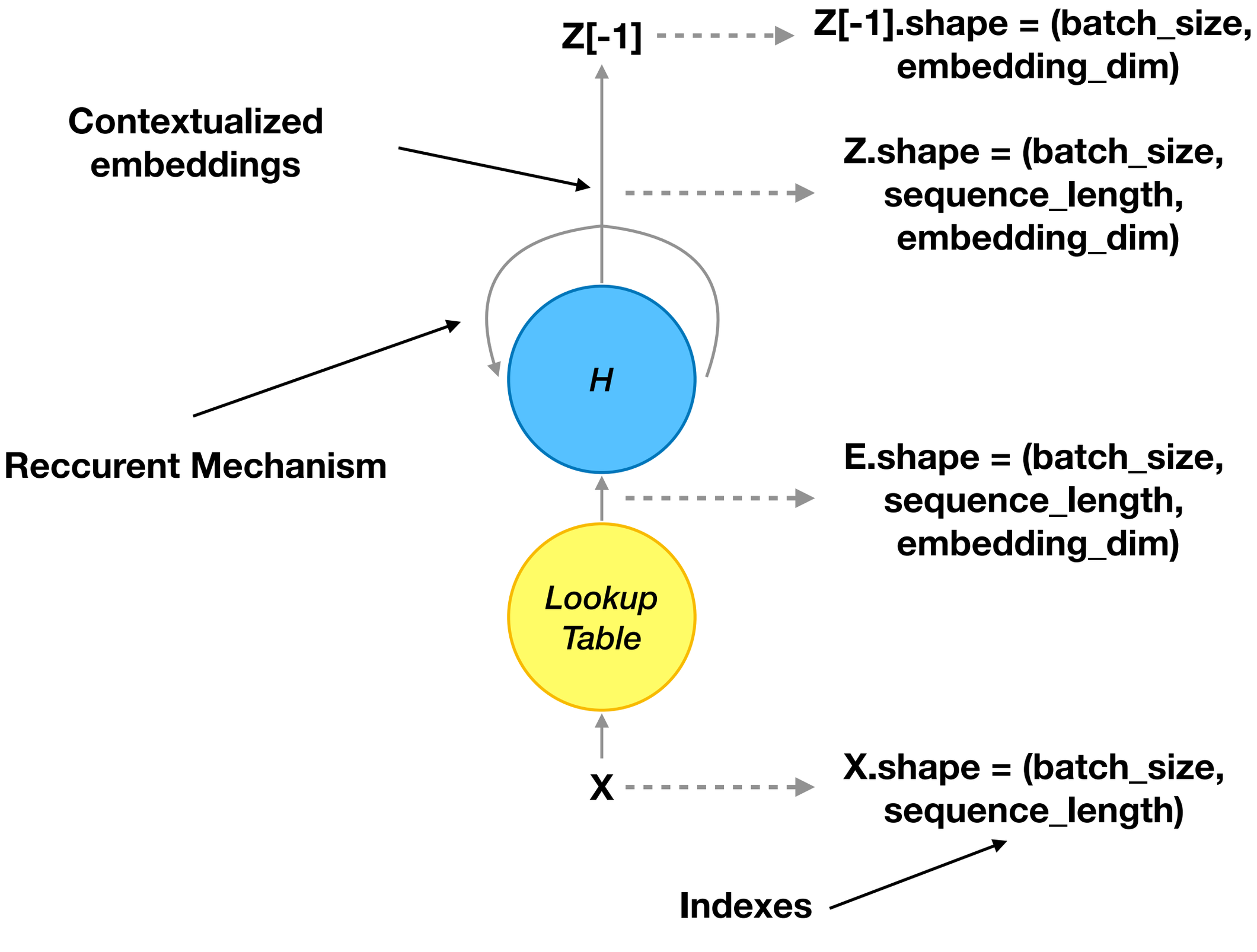
Neural Network



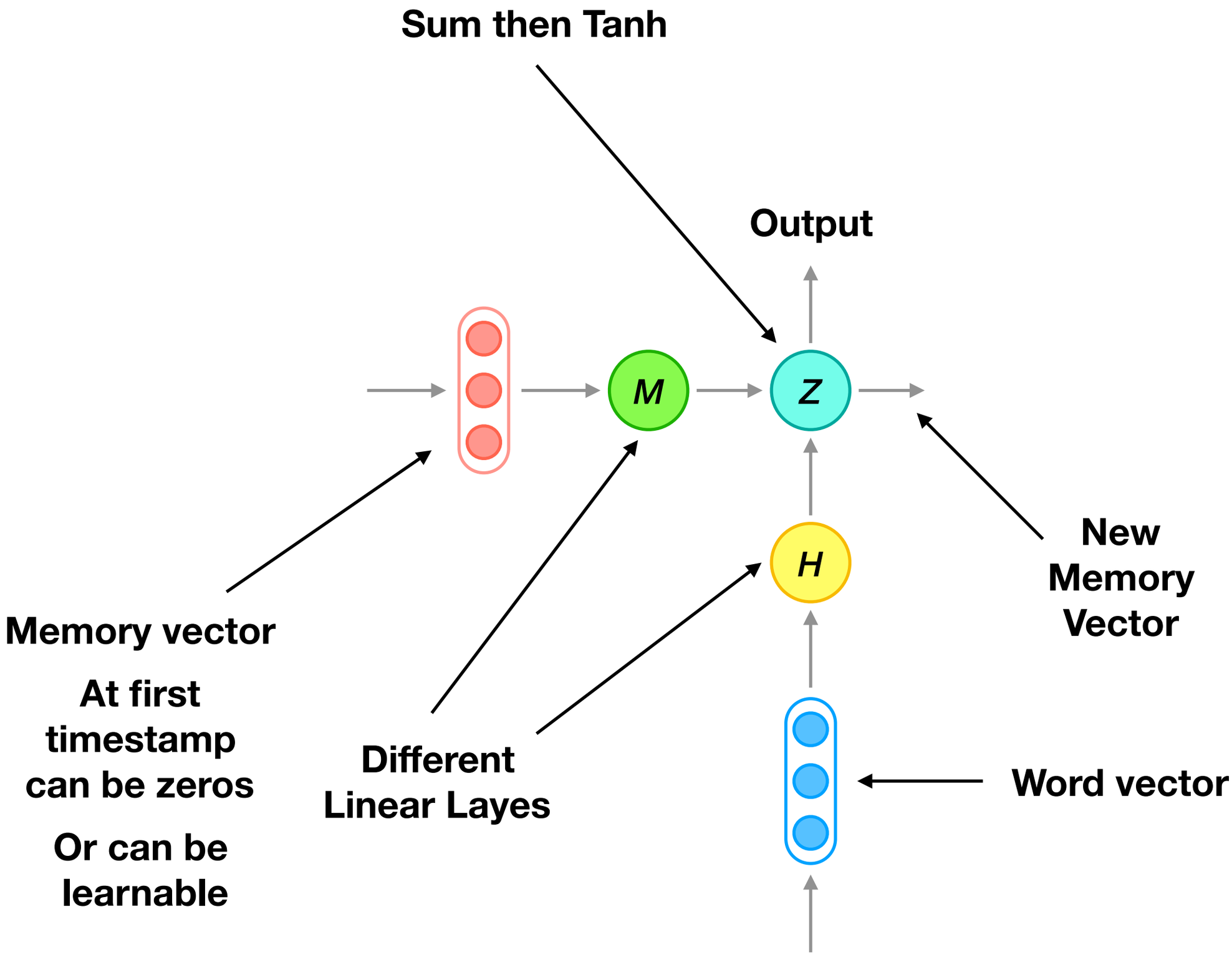
Neural Network



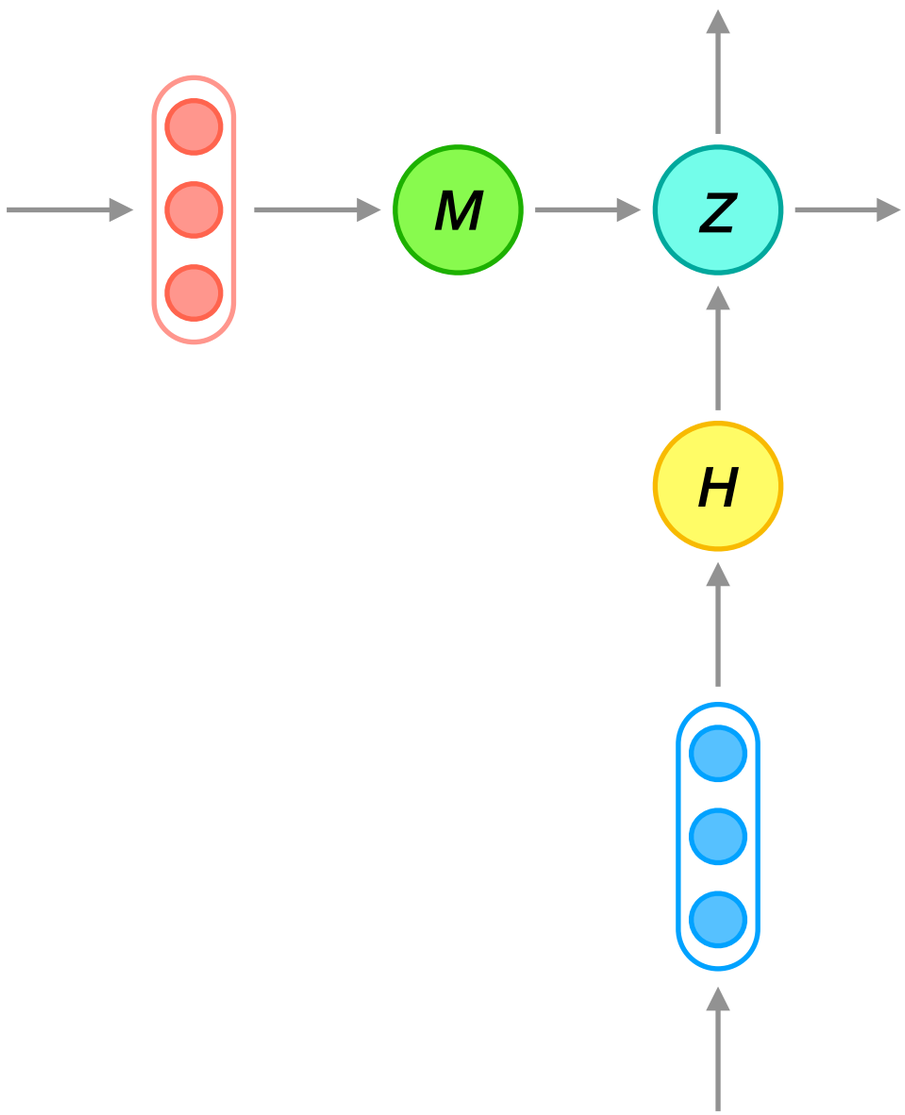
Recurrent Neural Network



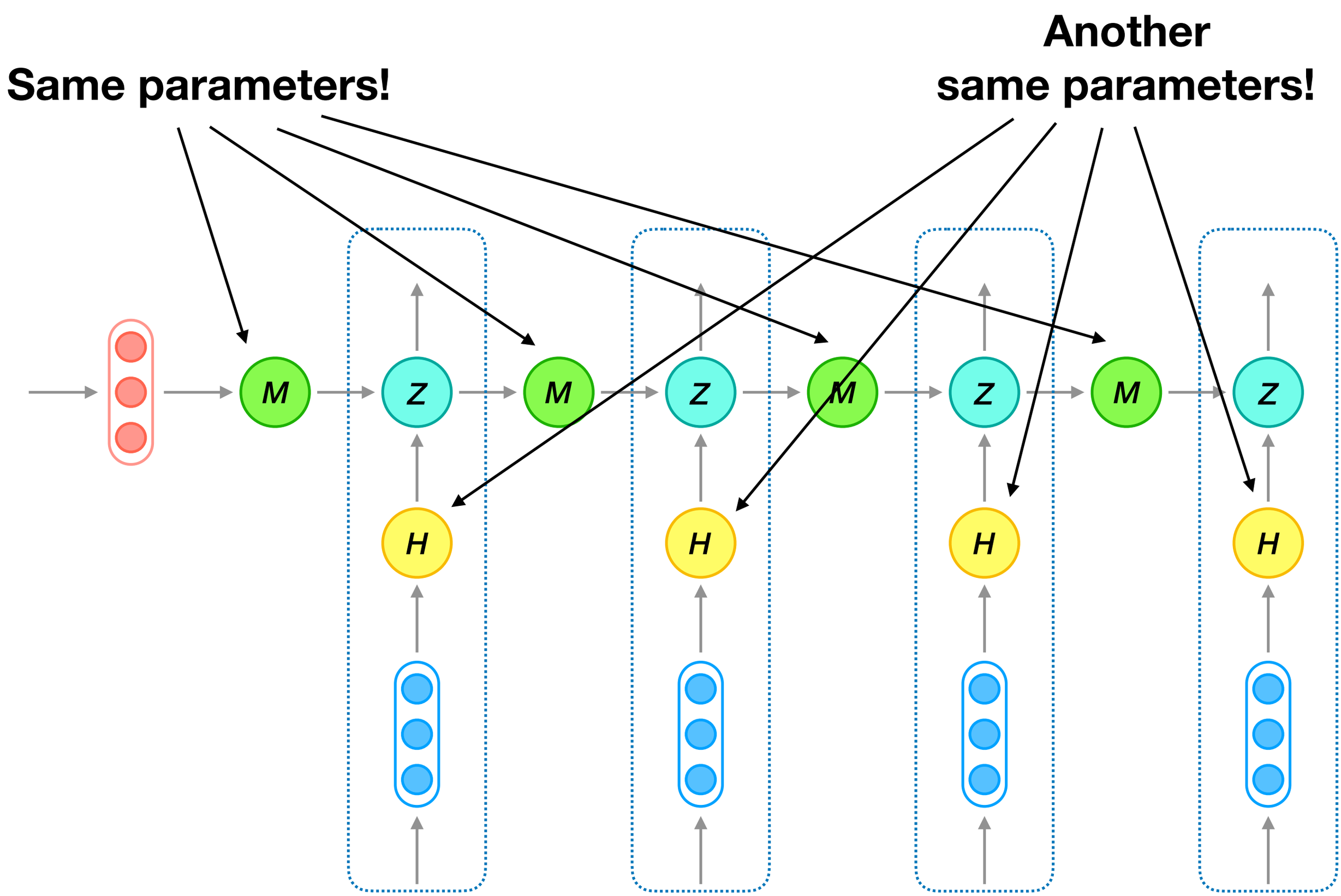
Reccurent Cell



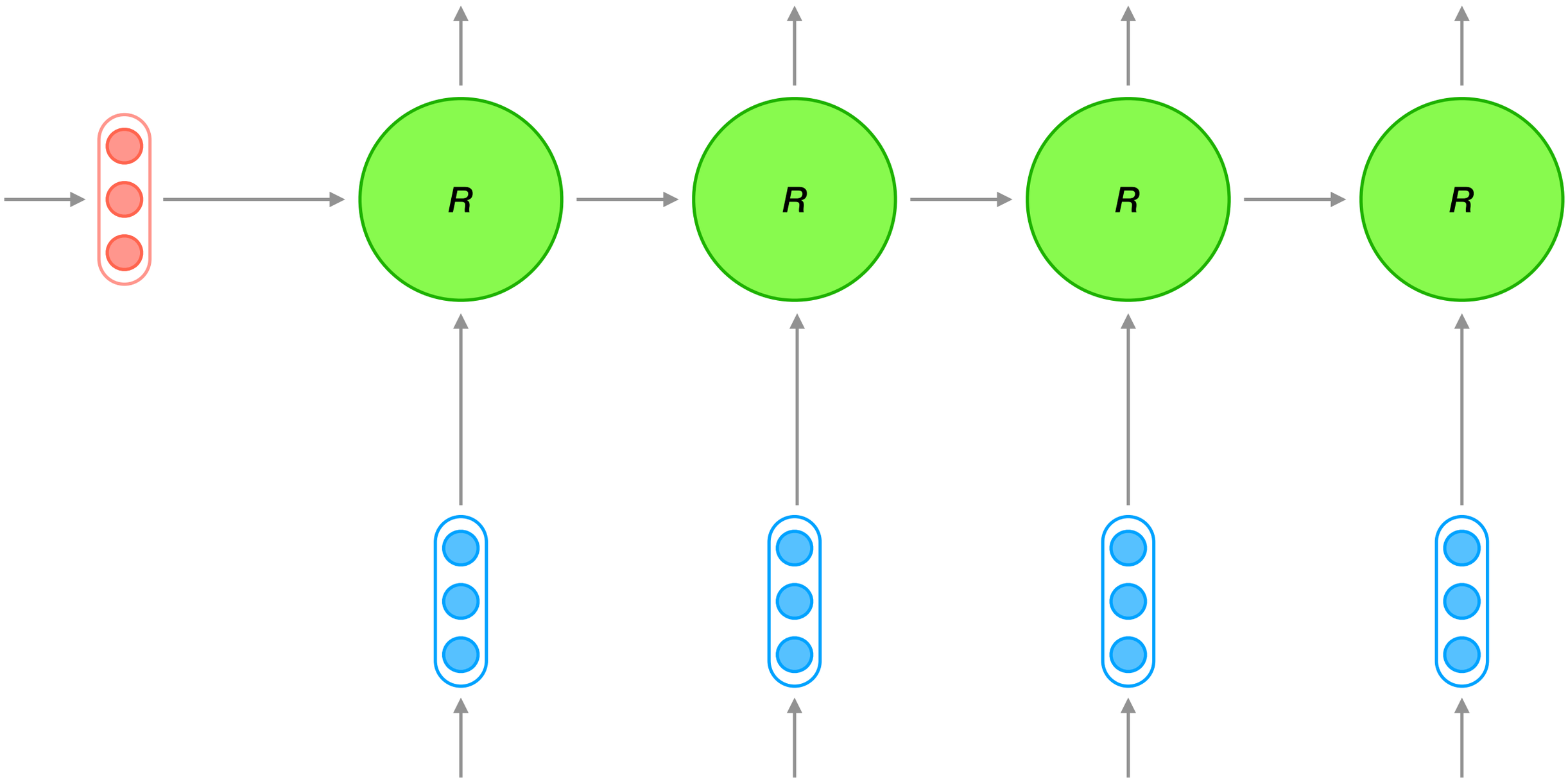
Reccurent Mechanism



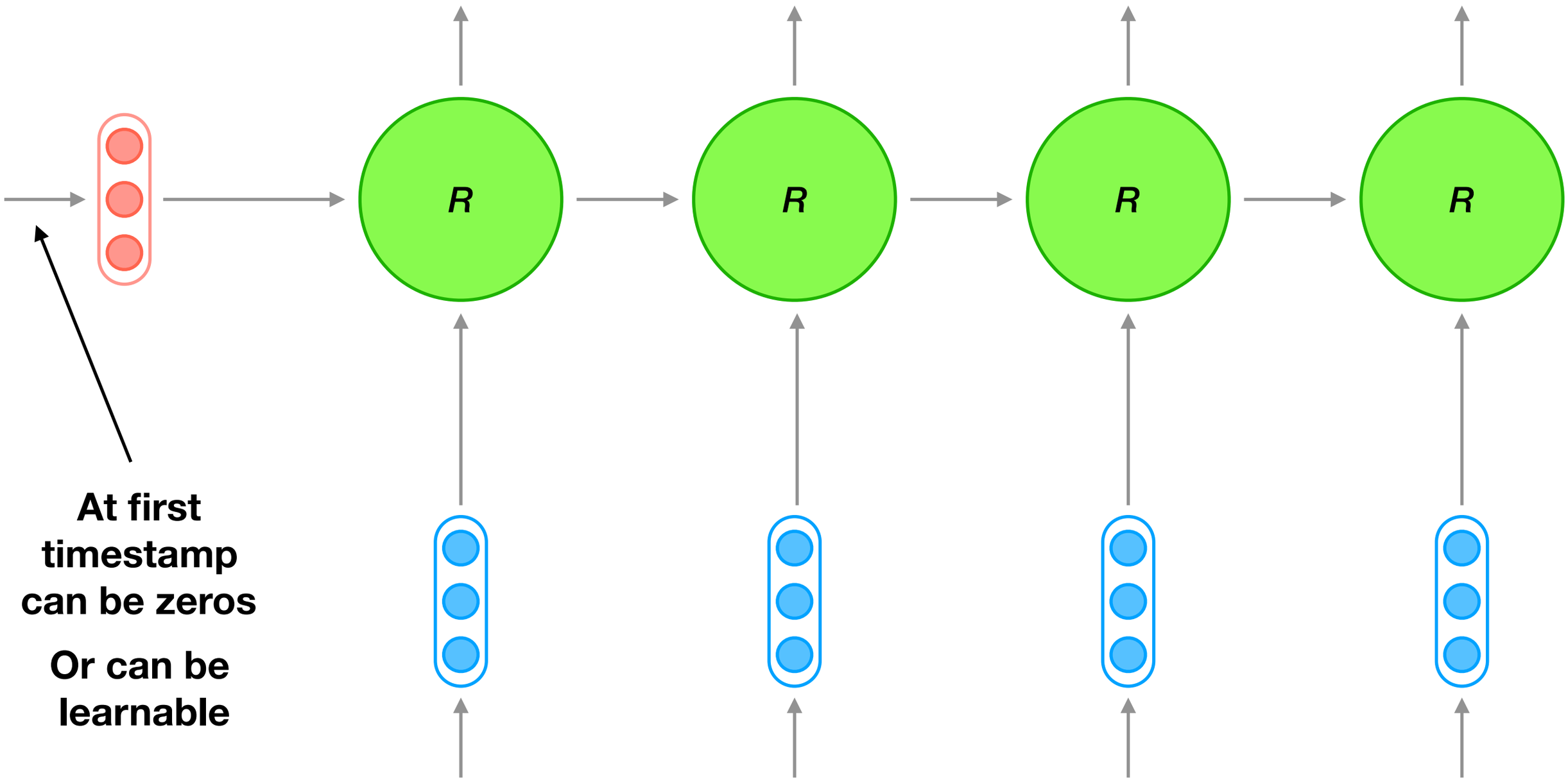
Reccurent Mechanism



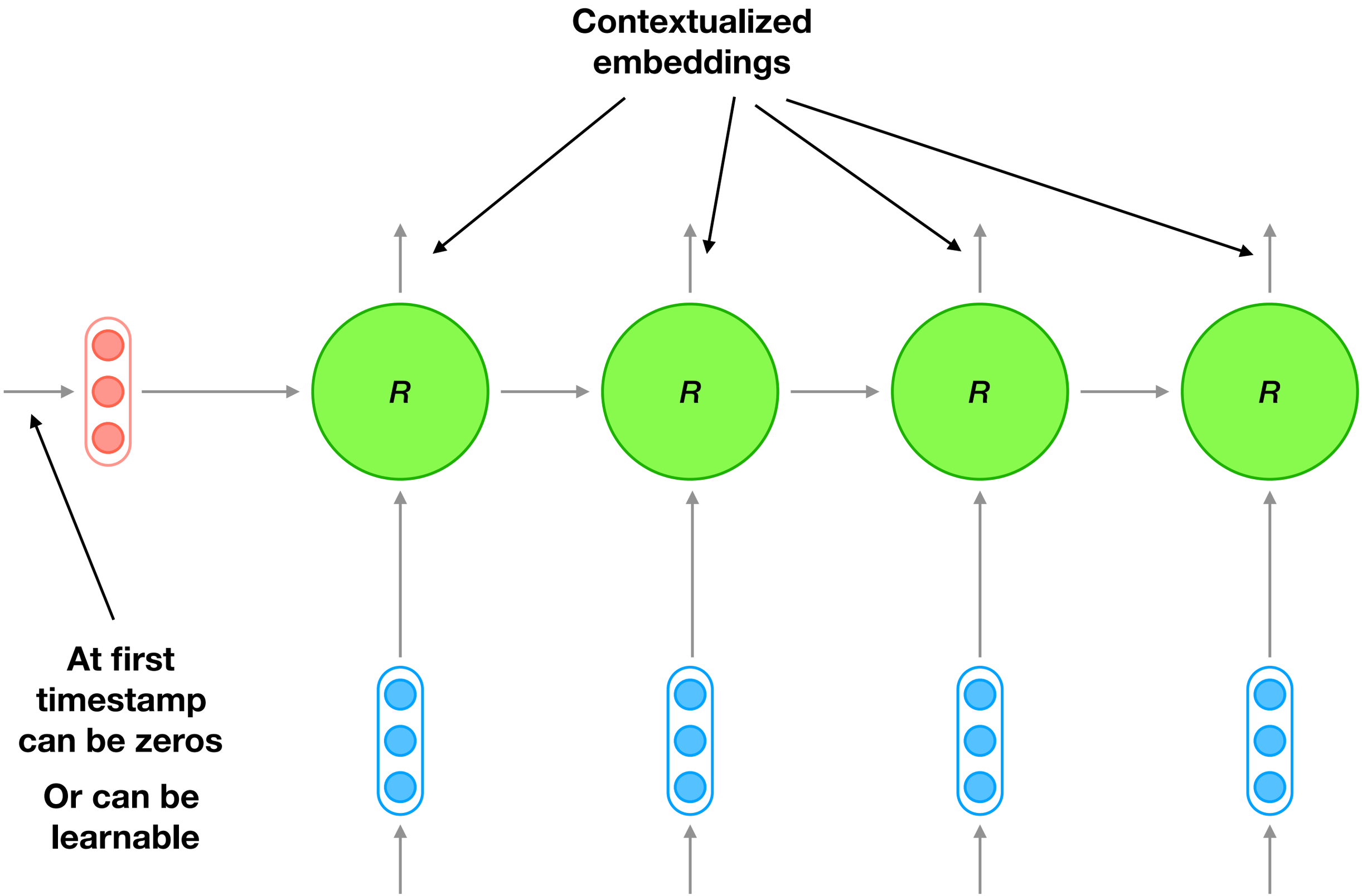
Reccurent Mechanism



Reccurent Mechanism

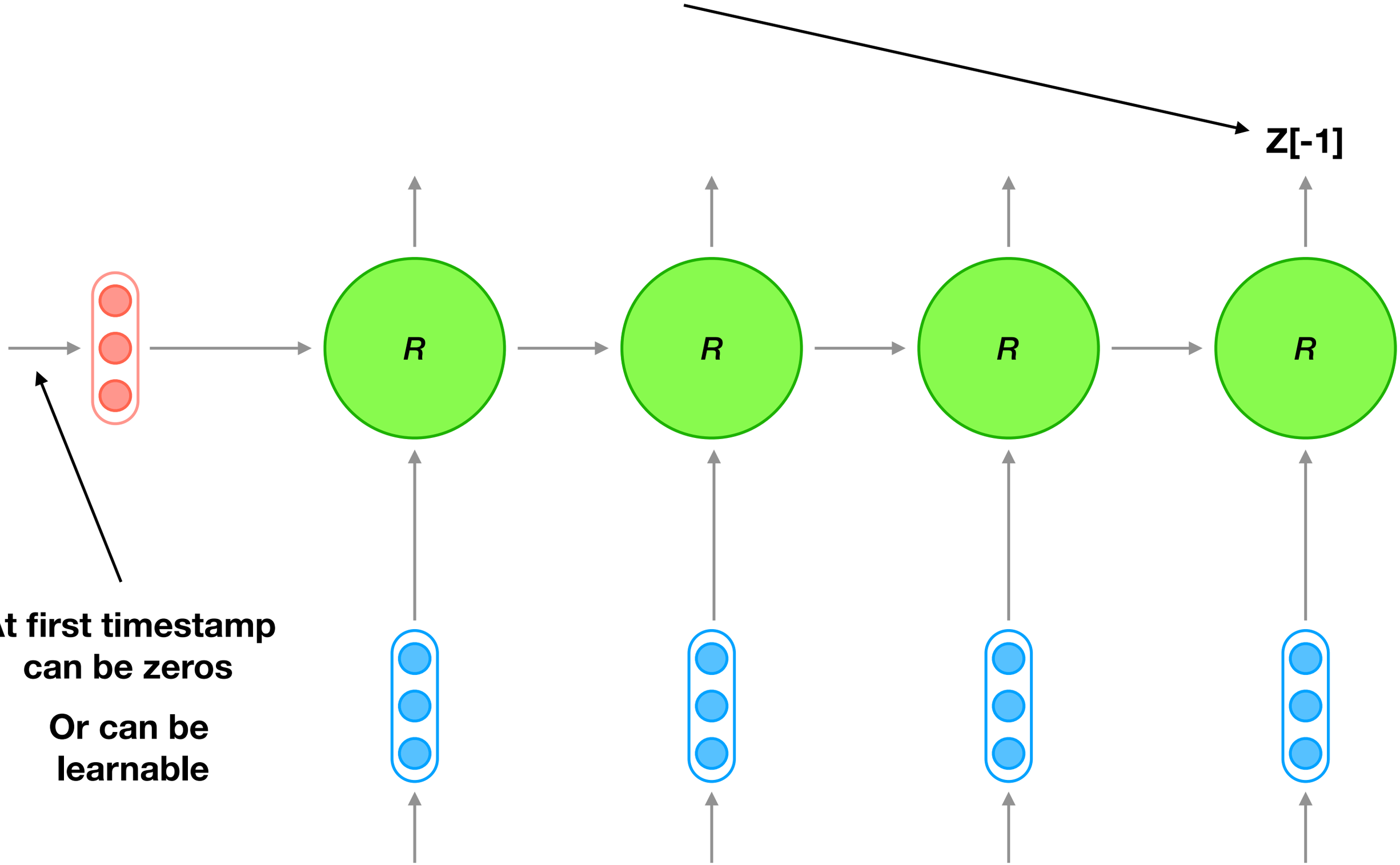


Reccurent Mechanism

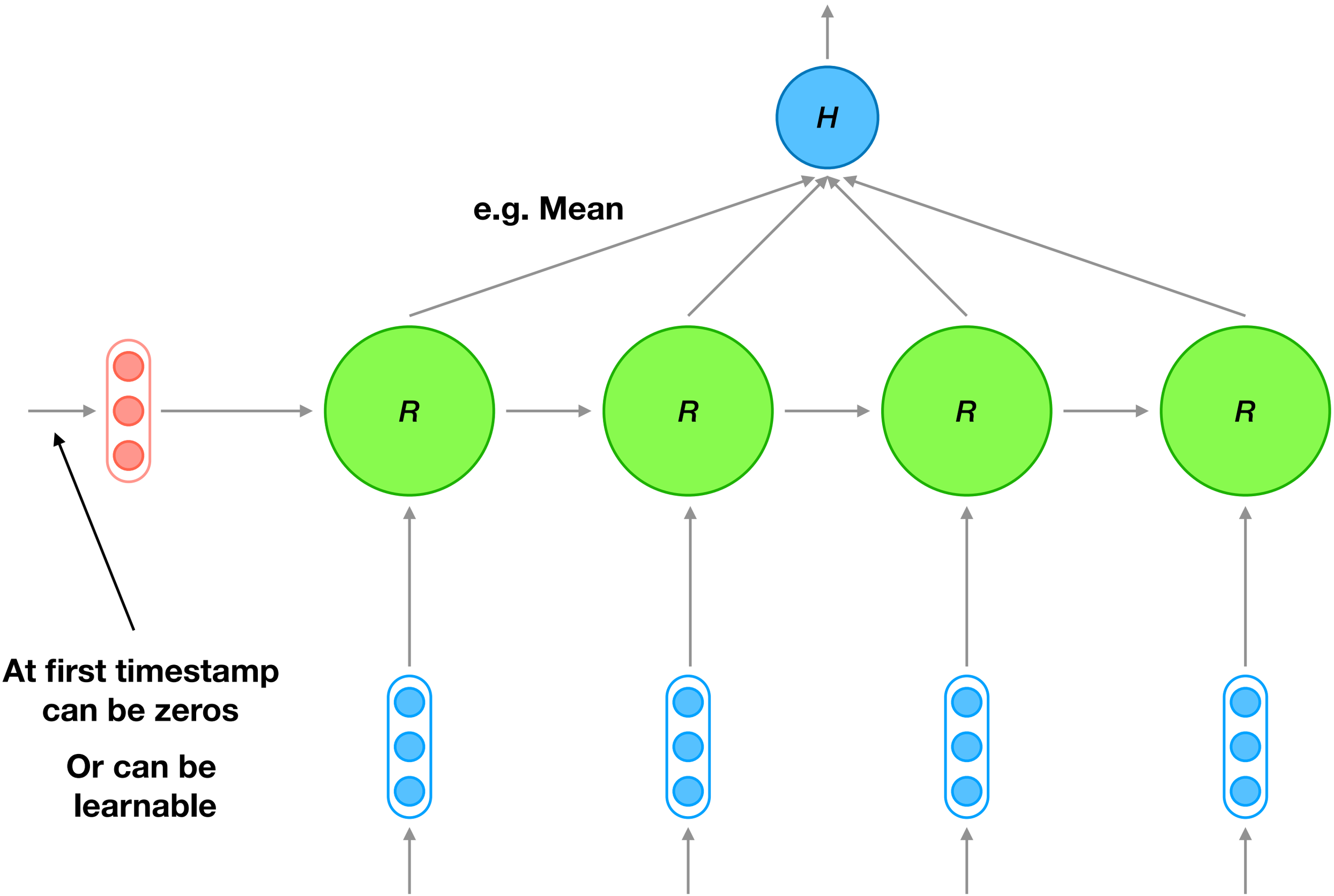


Reccurent Mechanism

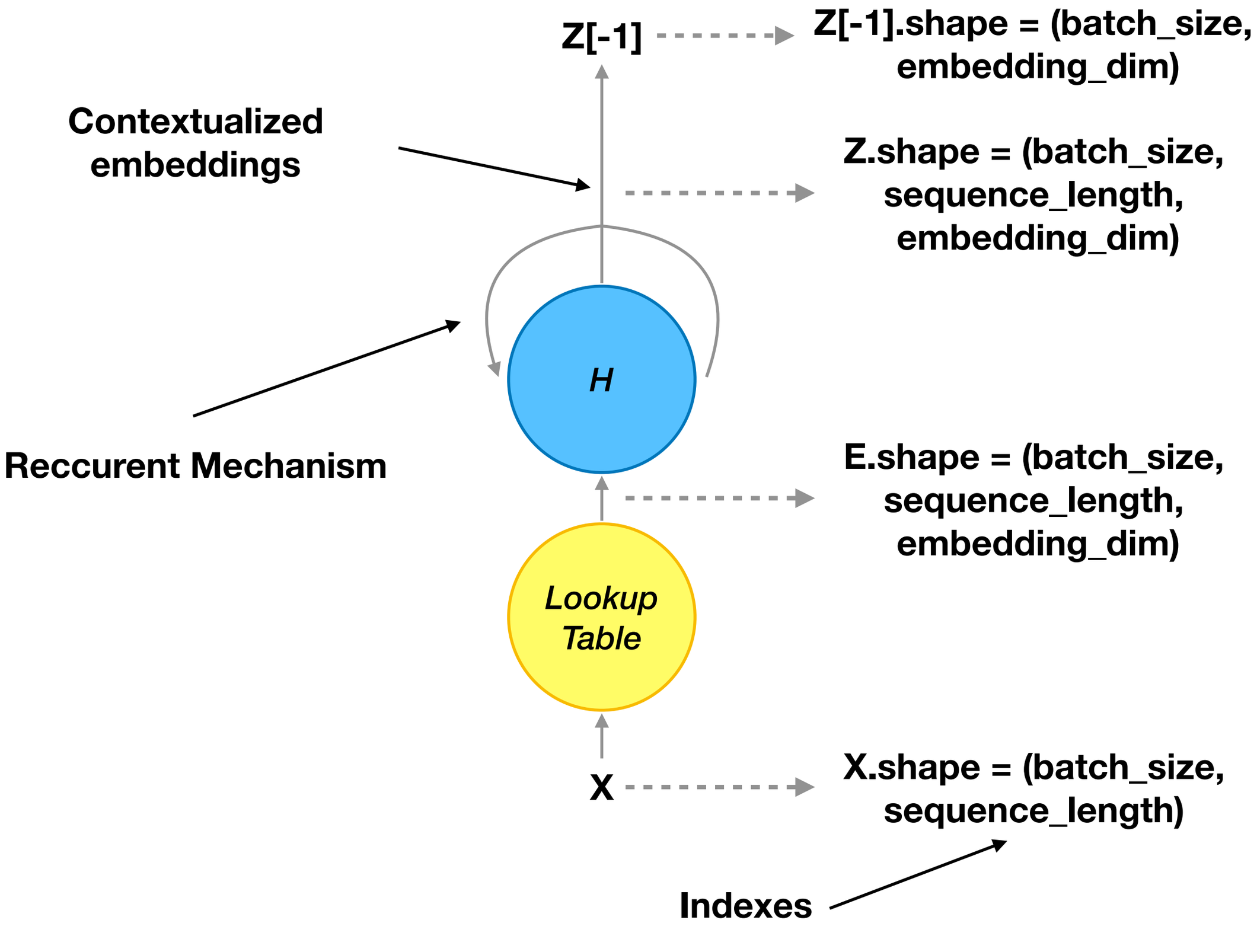
Now we can take last vector from sequence



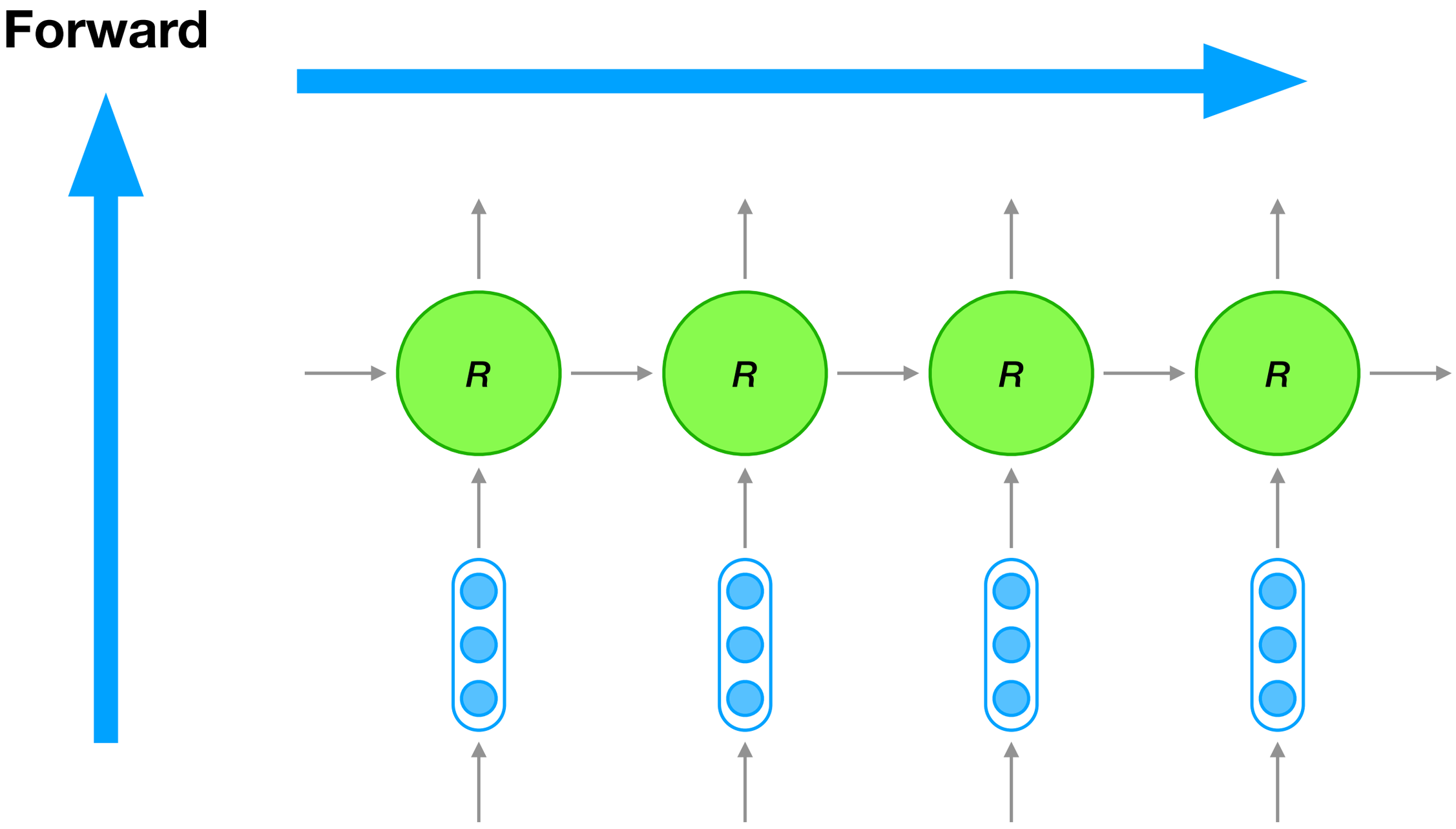
Reccurent Mechanism



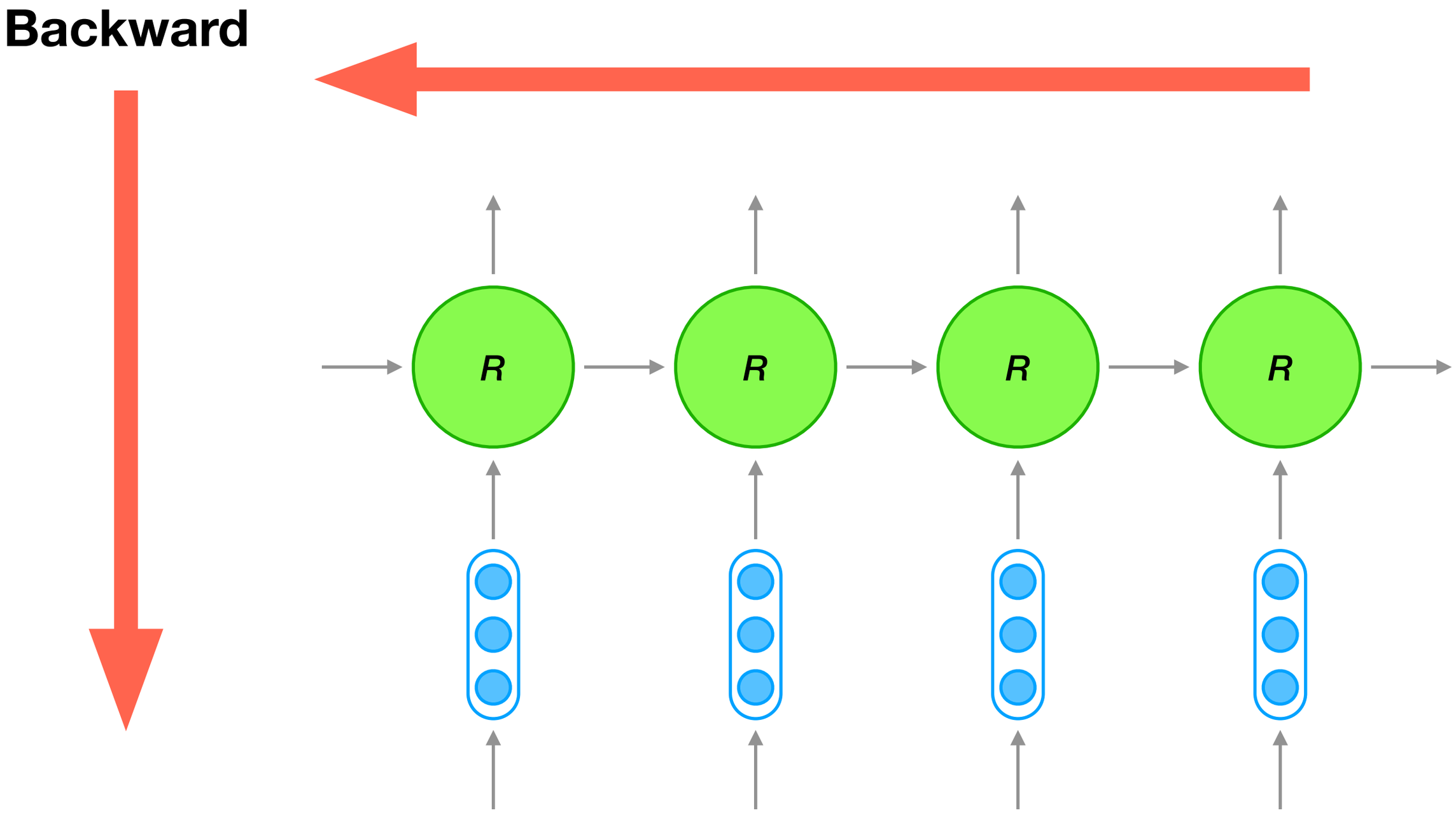
Recurrent Neural Network



BPTT

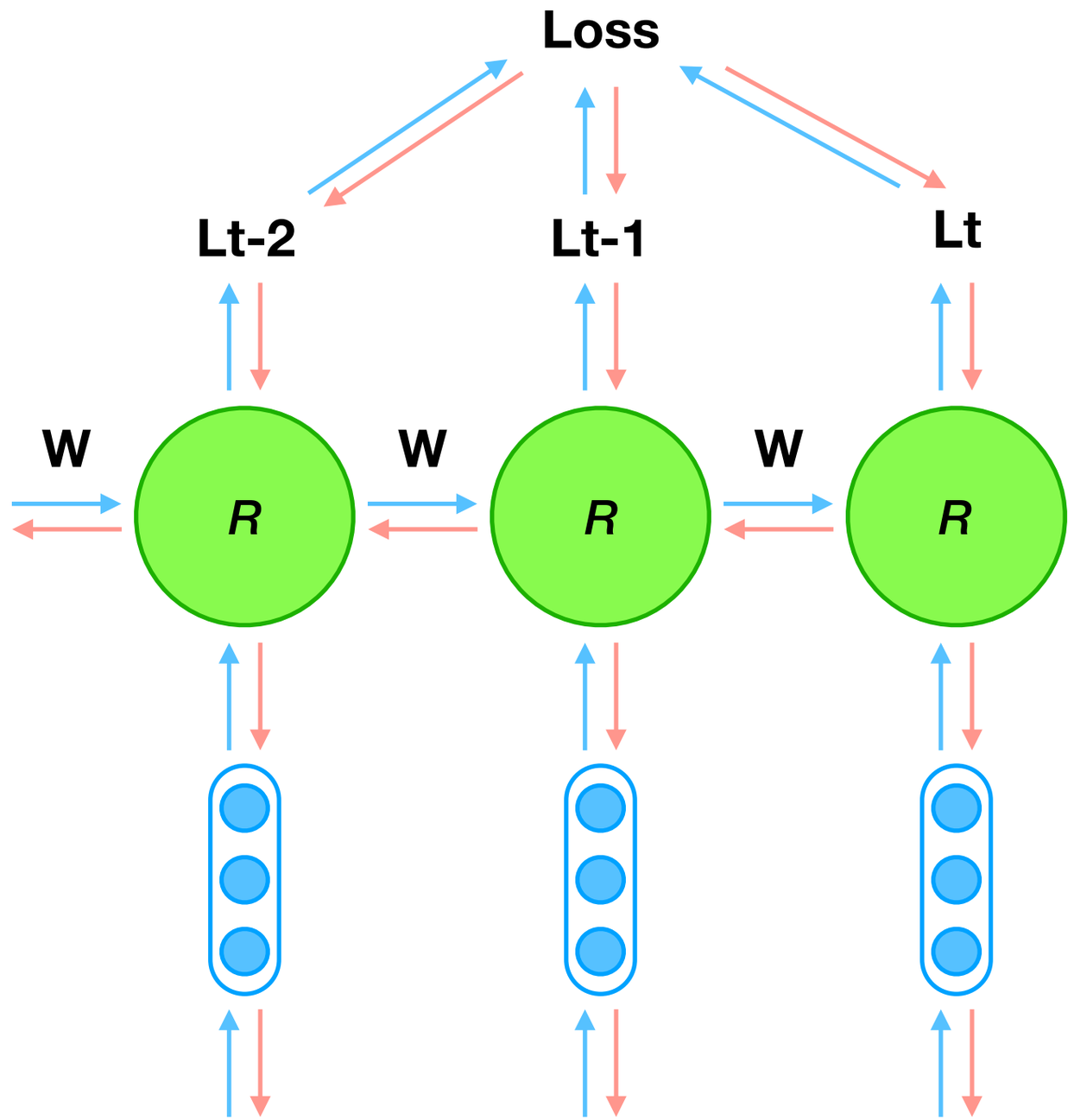


BPTT



BPTT

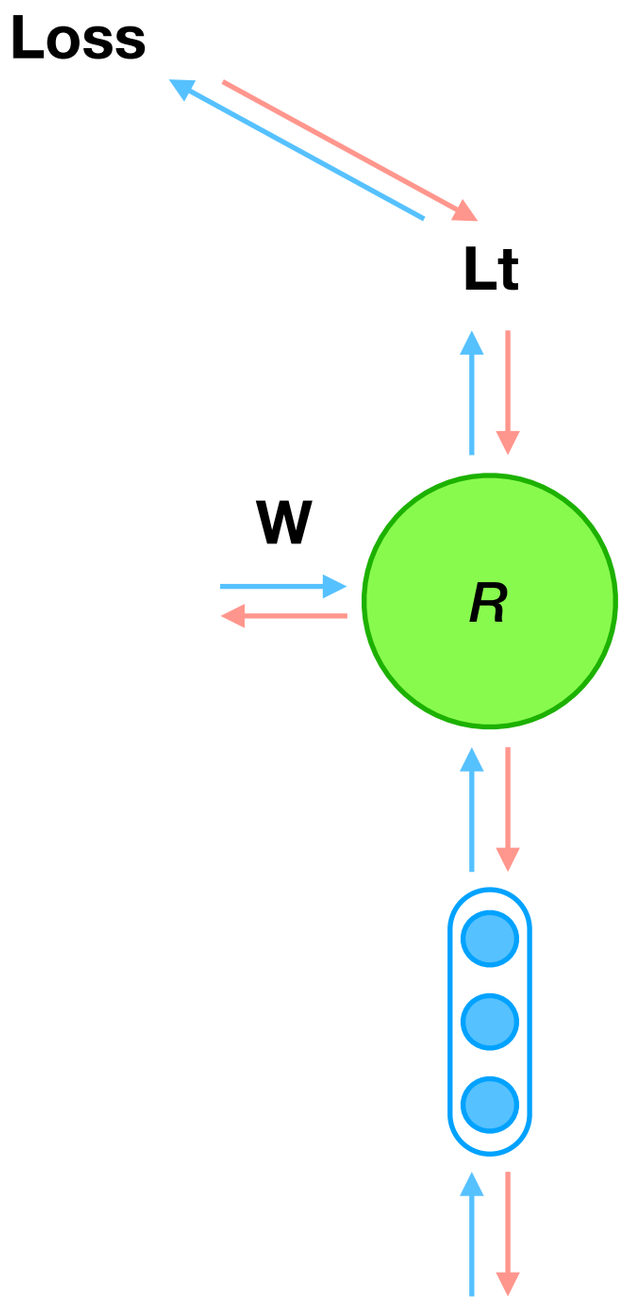
$$\frac{\partial L}{\partial W} = \sum_{i=0}^{T} \frac{\partial L_i}{\partial W}$$



BPTT

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

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



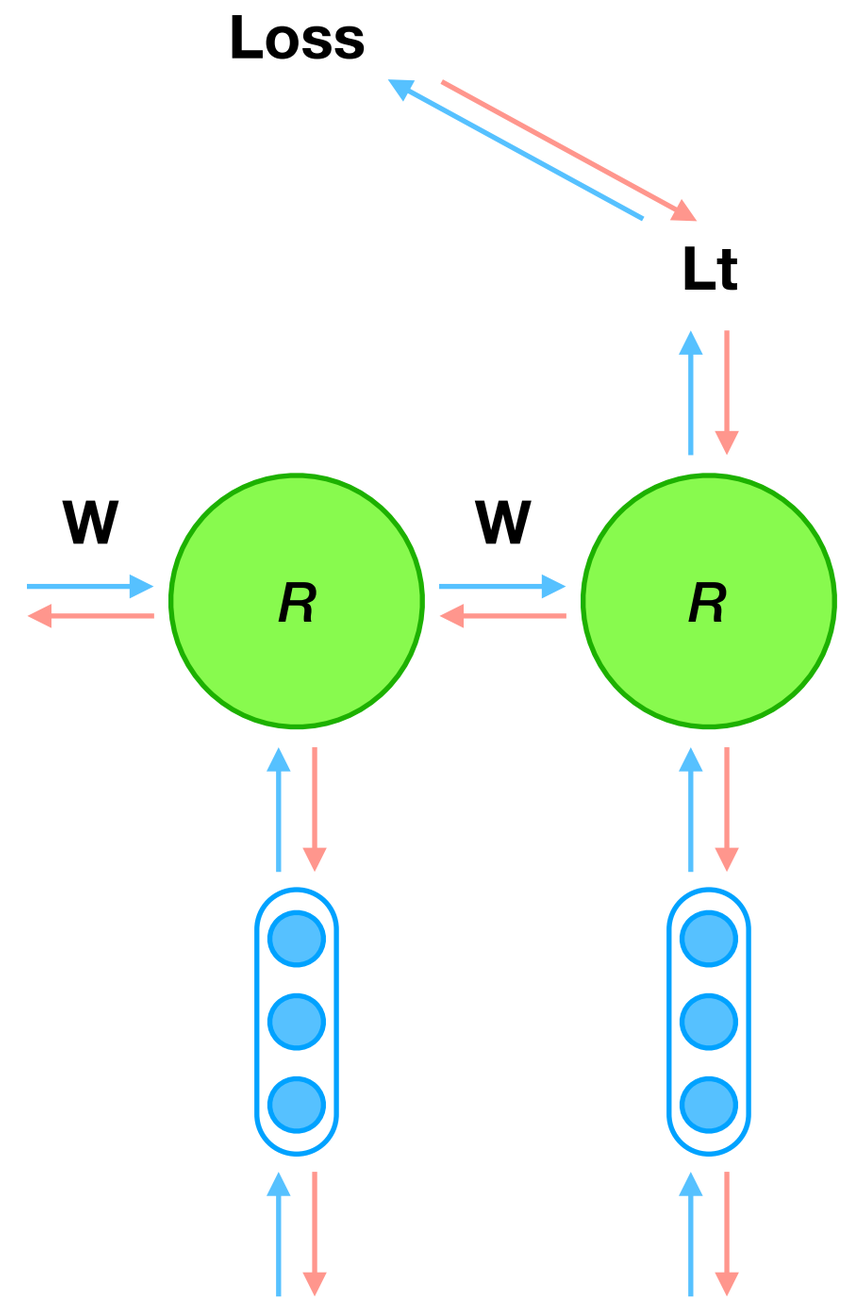
BPTT

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

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

$$h_t = f_h(V x_t + W h_{t-1} + b_h)$$

This is NOT the only dependence!



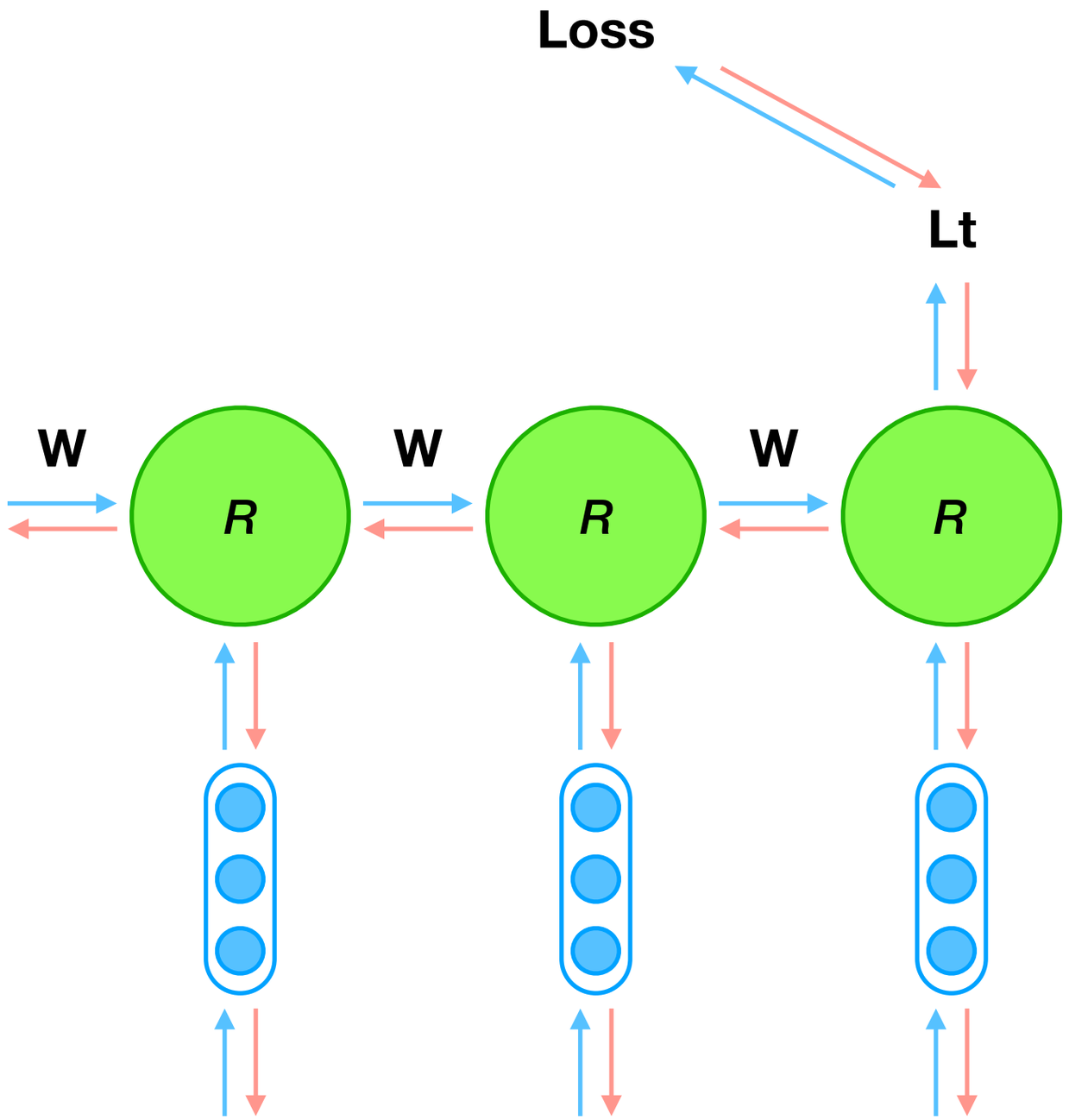
BPTT

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

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

$$h_t = f_h(V x_t + W h_{t-1} + b_h)$$

This is NOT the only dependence!



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

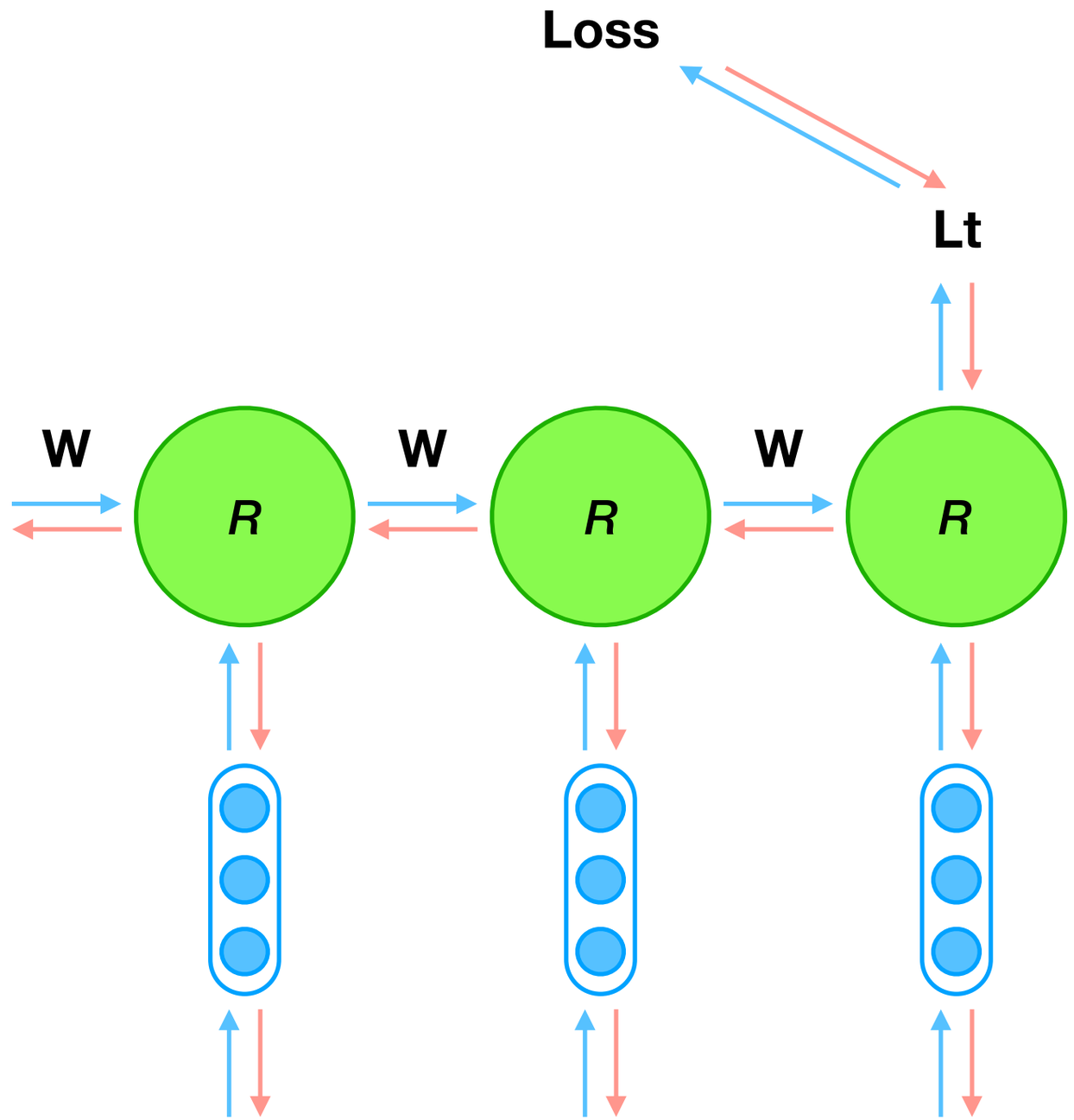
BPTT

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

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

$$h_t = f_h(V x_t + W h_{t-1} + b_h)$$

This is NOT the only dependence!



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

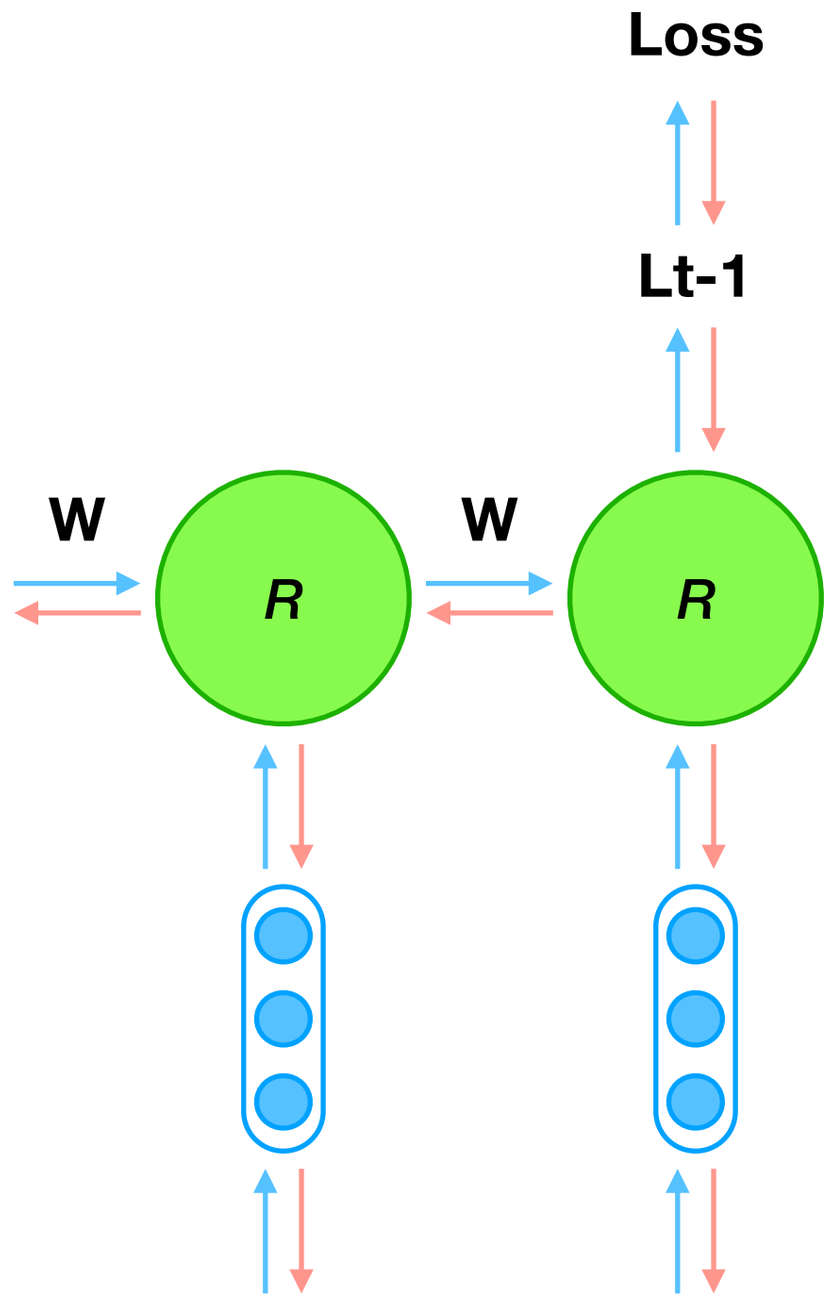
BPTT

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

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

$$h_t = f_h(V x_t + W h_{t-1} + b_h)$$

This is NOT the only dependence!



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

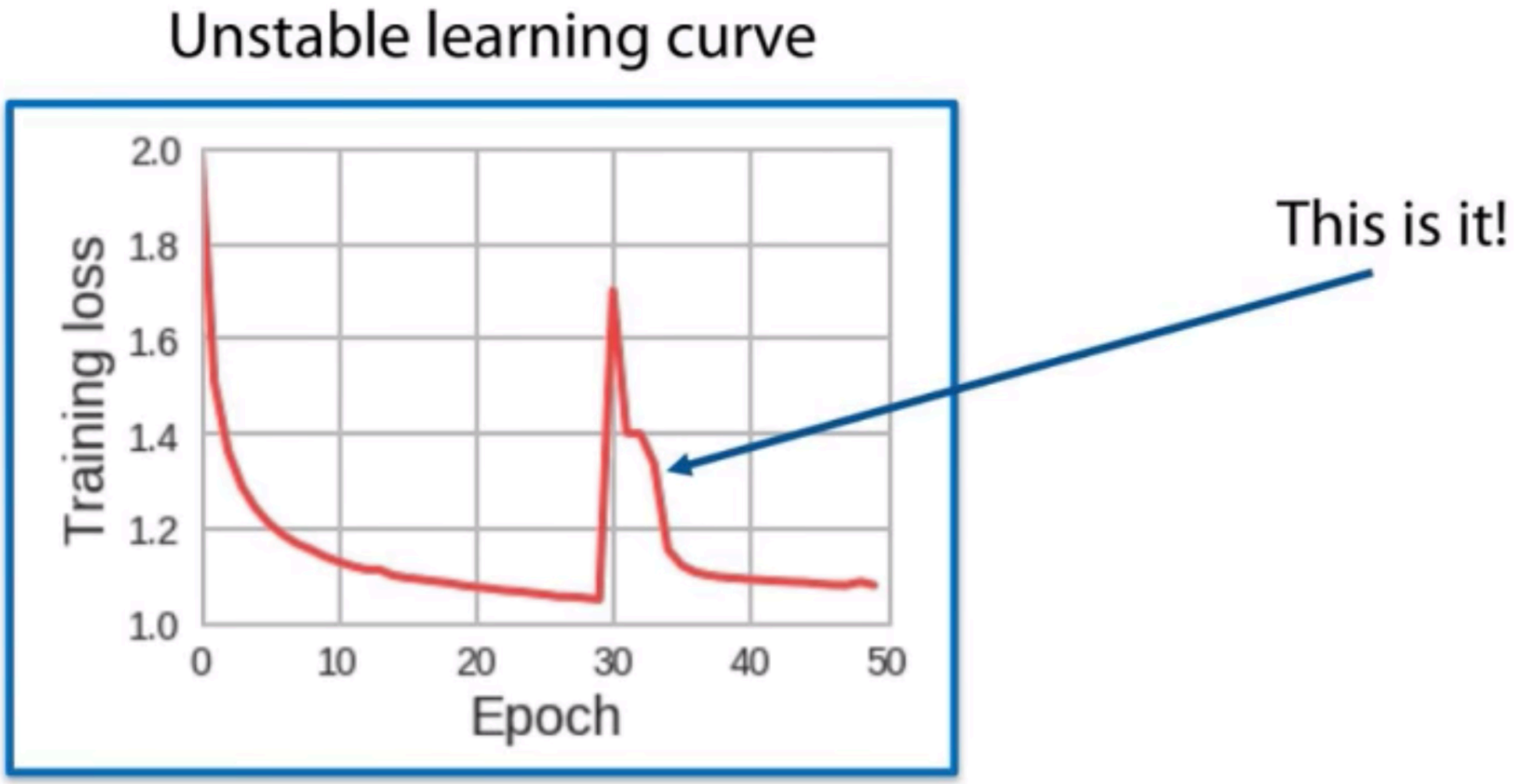
Gradients

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

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

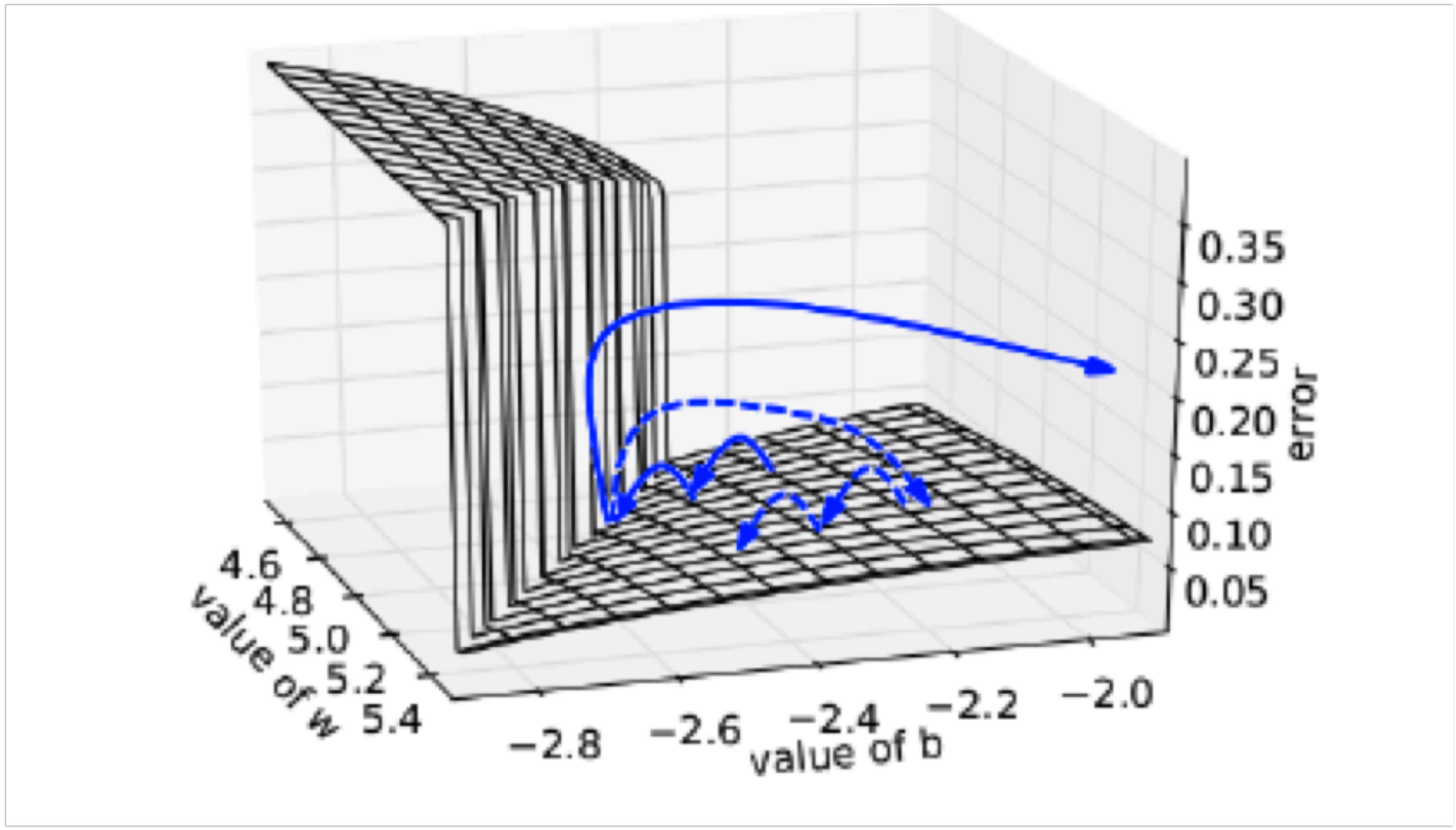
$$\left\| \frac{\partial h_i}{\partial h_{i-1}} \right\|_2 > 1 \quad \Rightarrow \quad \text{Exploding gradients}$$

Exploding gradients



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

Exploding gradients



Gradient Clipping

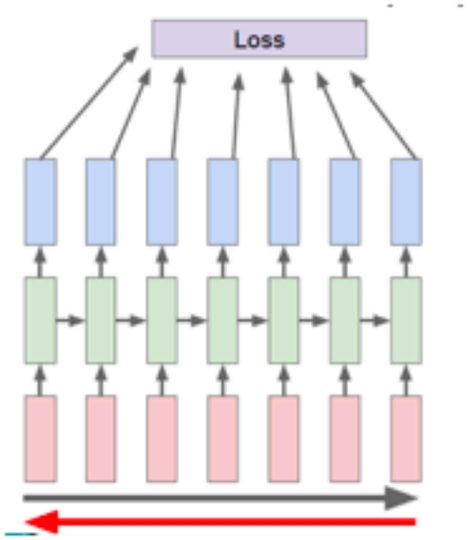
Gradient $g = \frac{\partial L}{\partial \theta}$, θ - all the network parameters

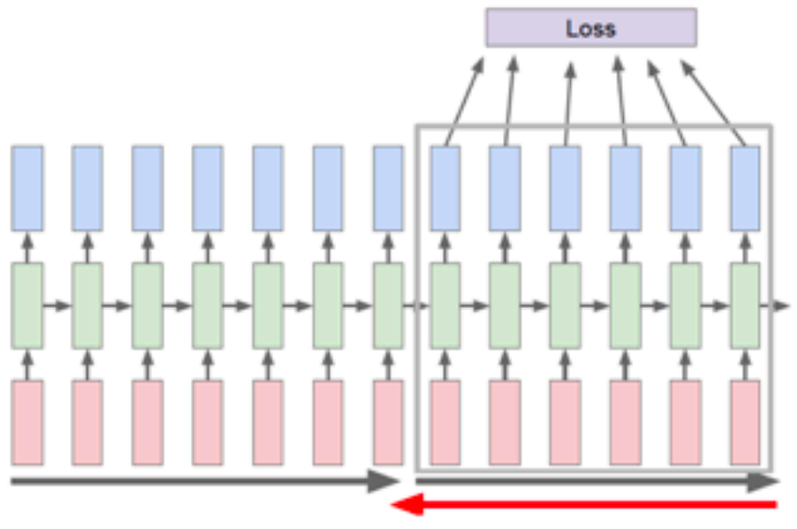
If $||g||$> threshold:

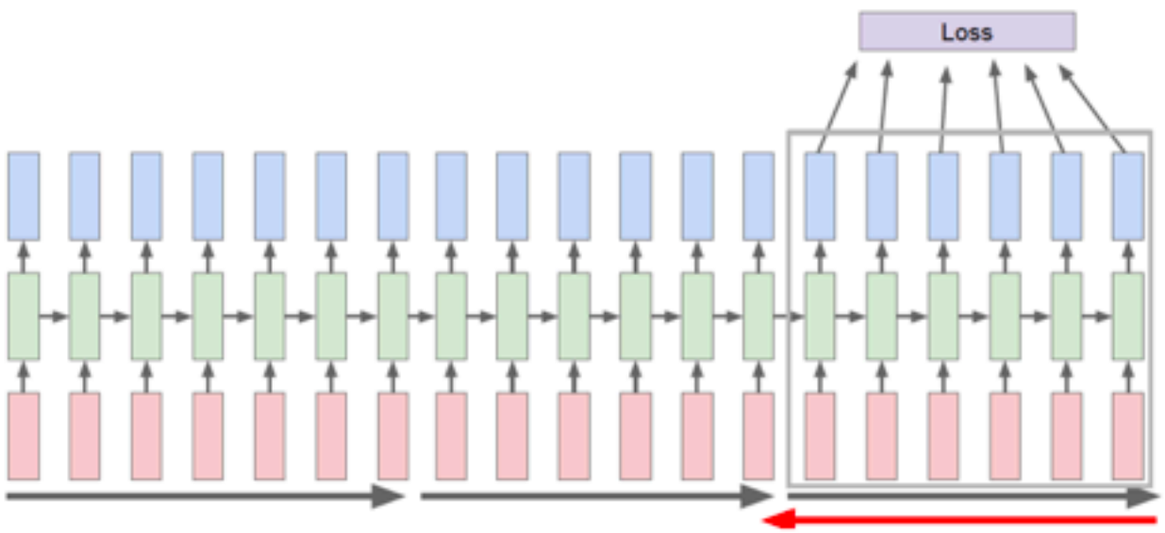
$$g \leftarrow \frac{threshold}{\|g\|} g$$

Simple but still very effective!

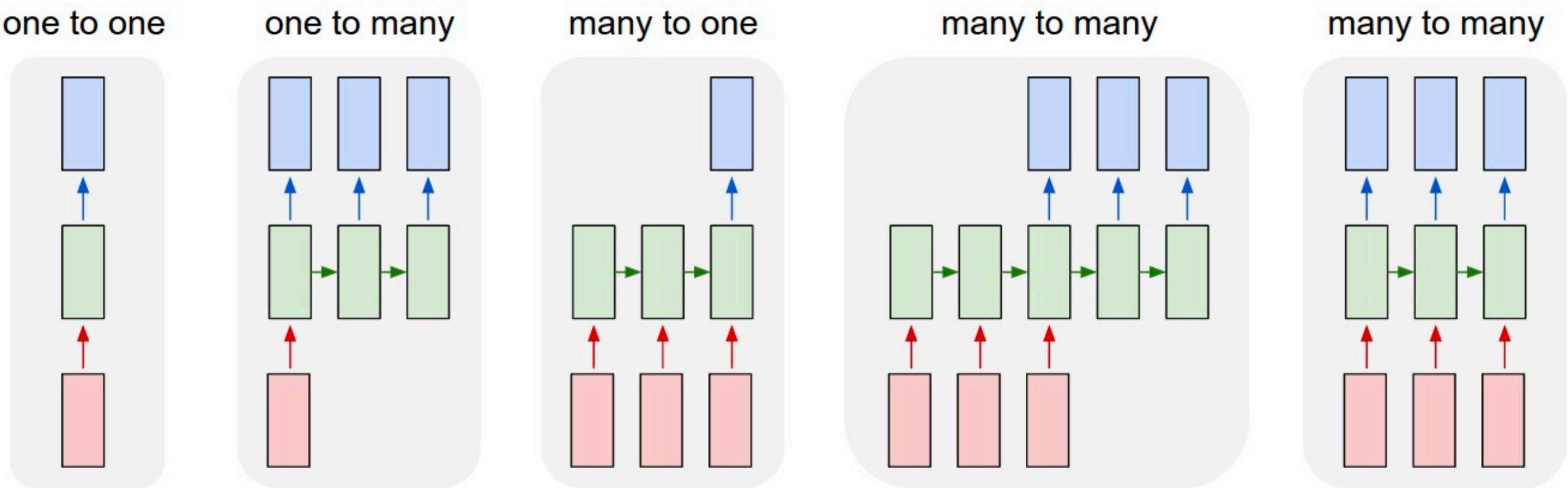
Truncated BPTT



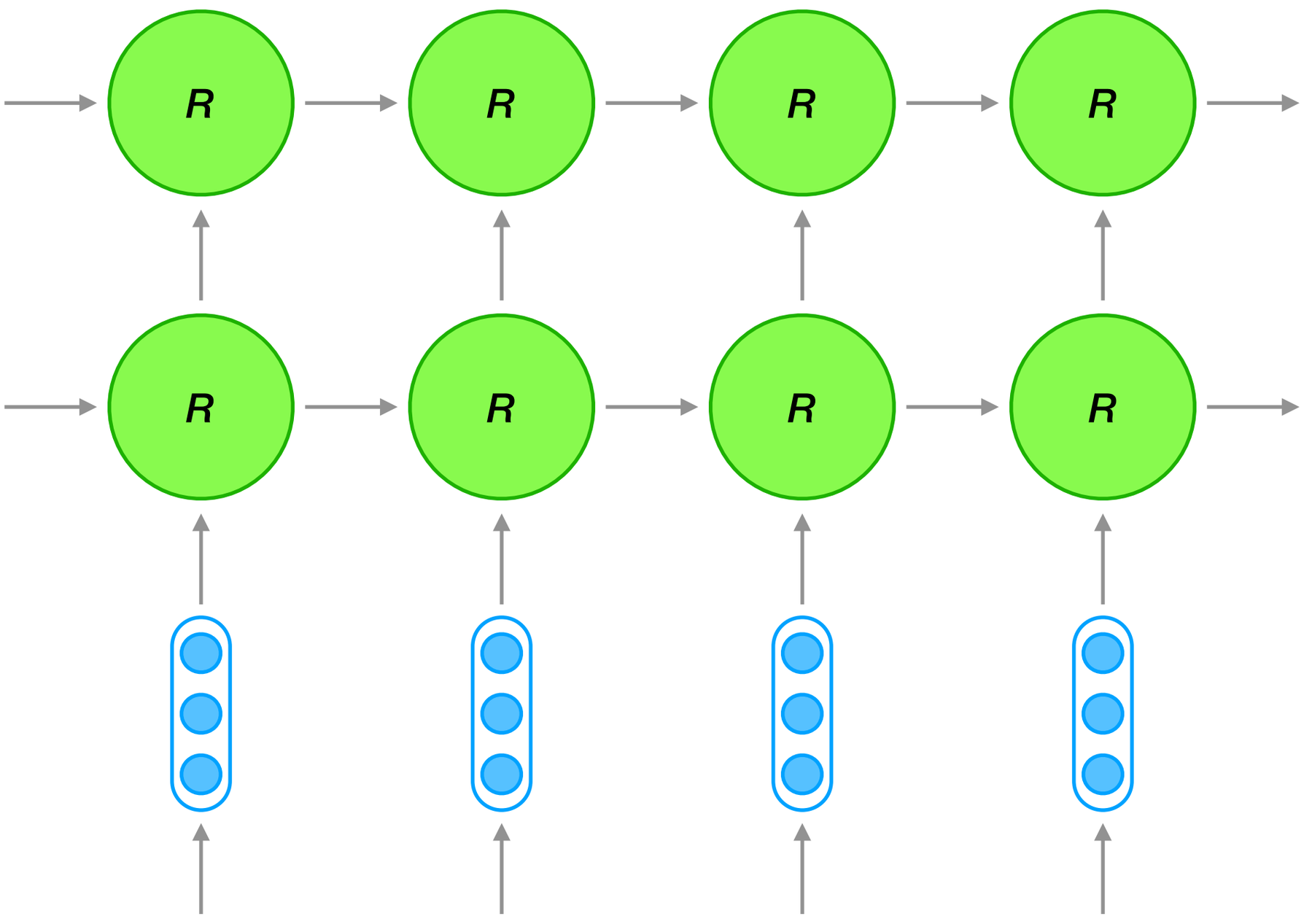




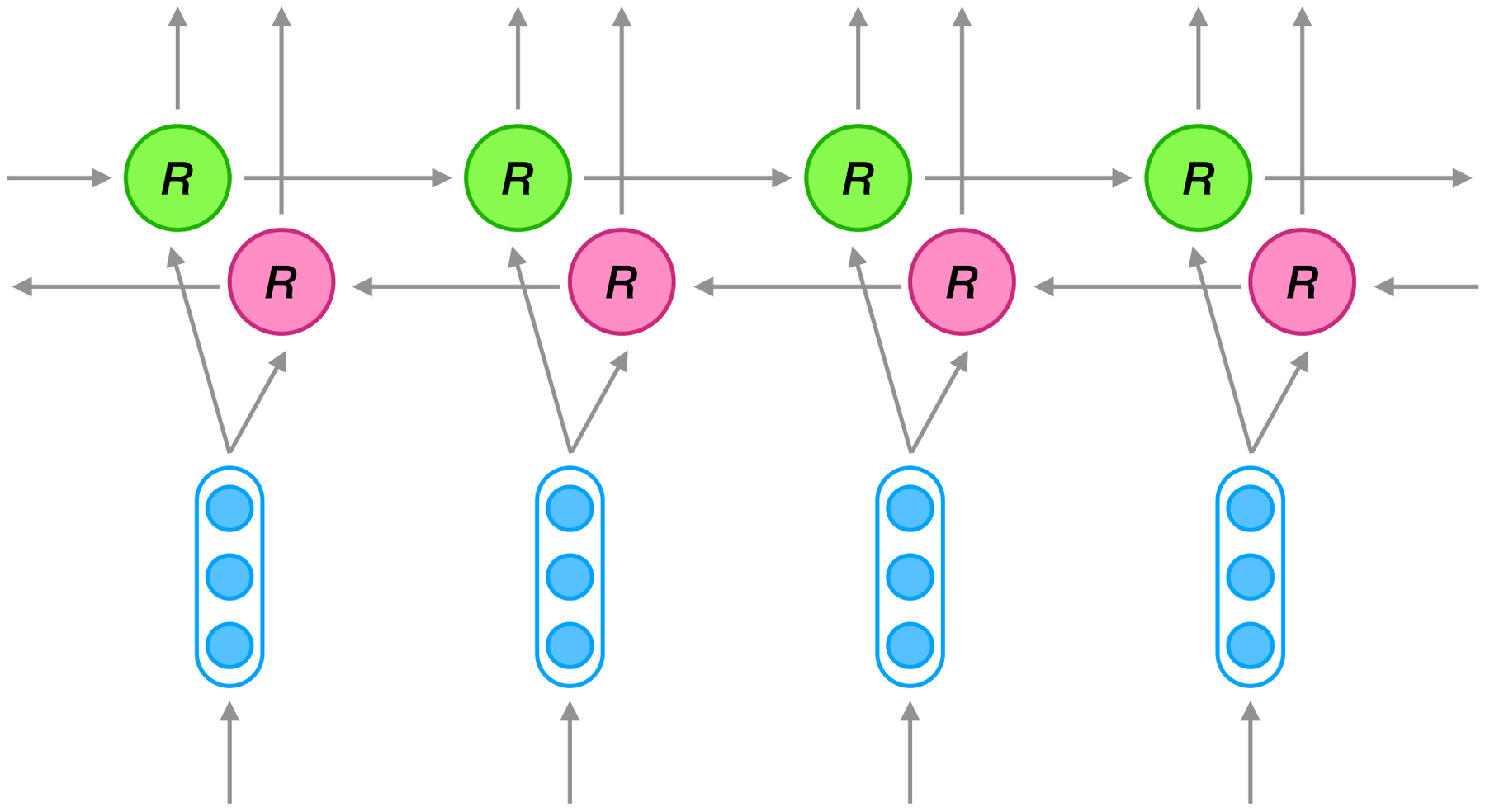
Recurrent Neural Network



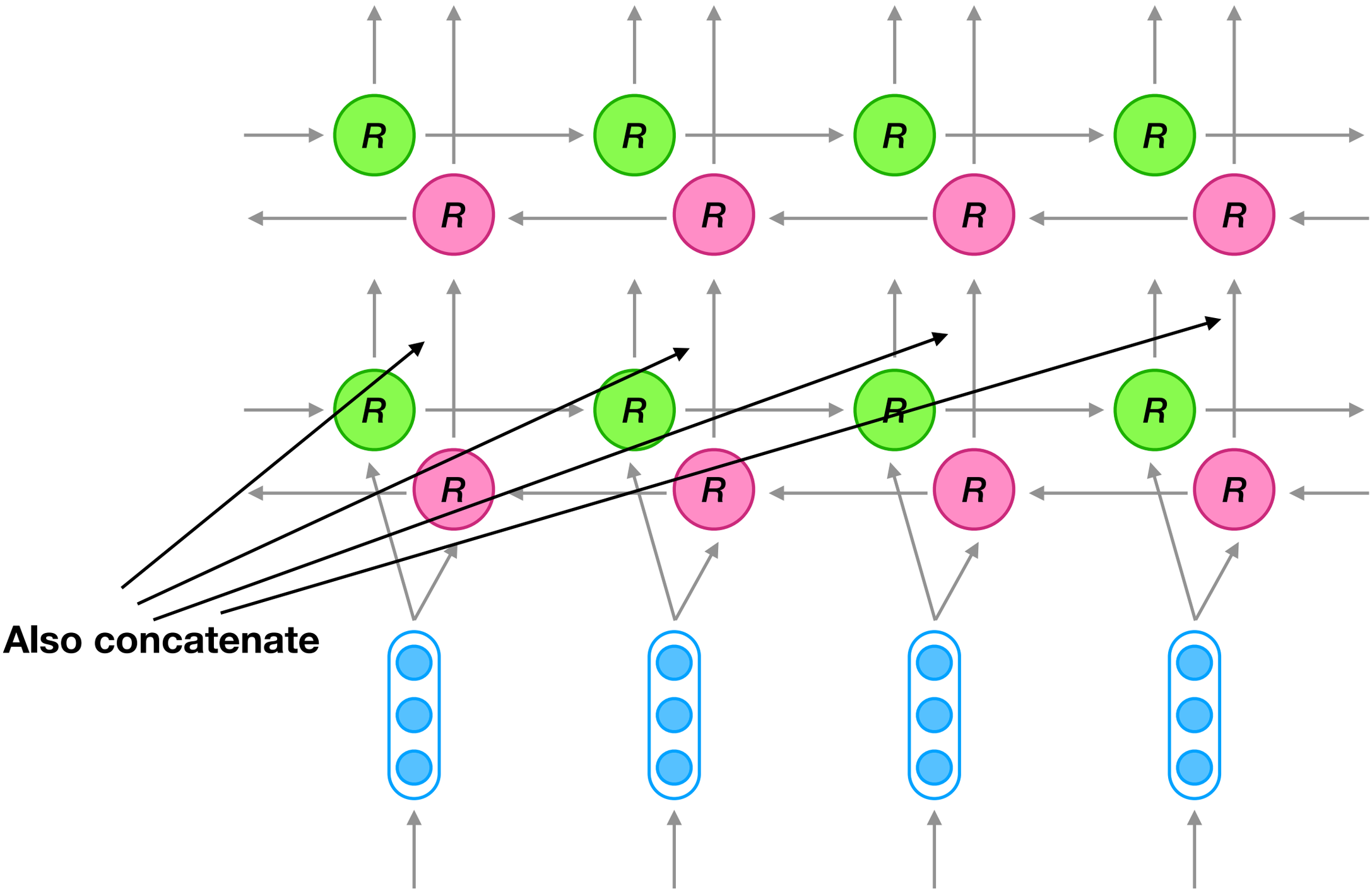
Deeper



Bidirectional



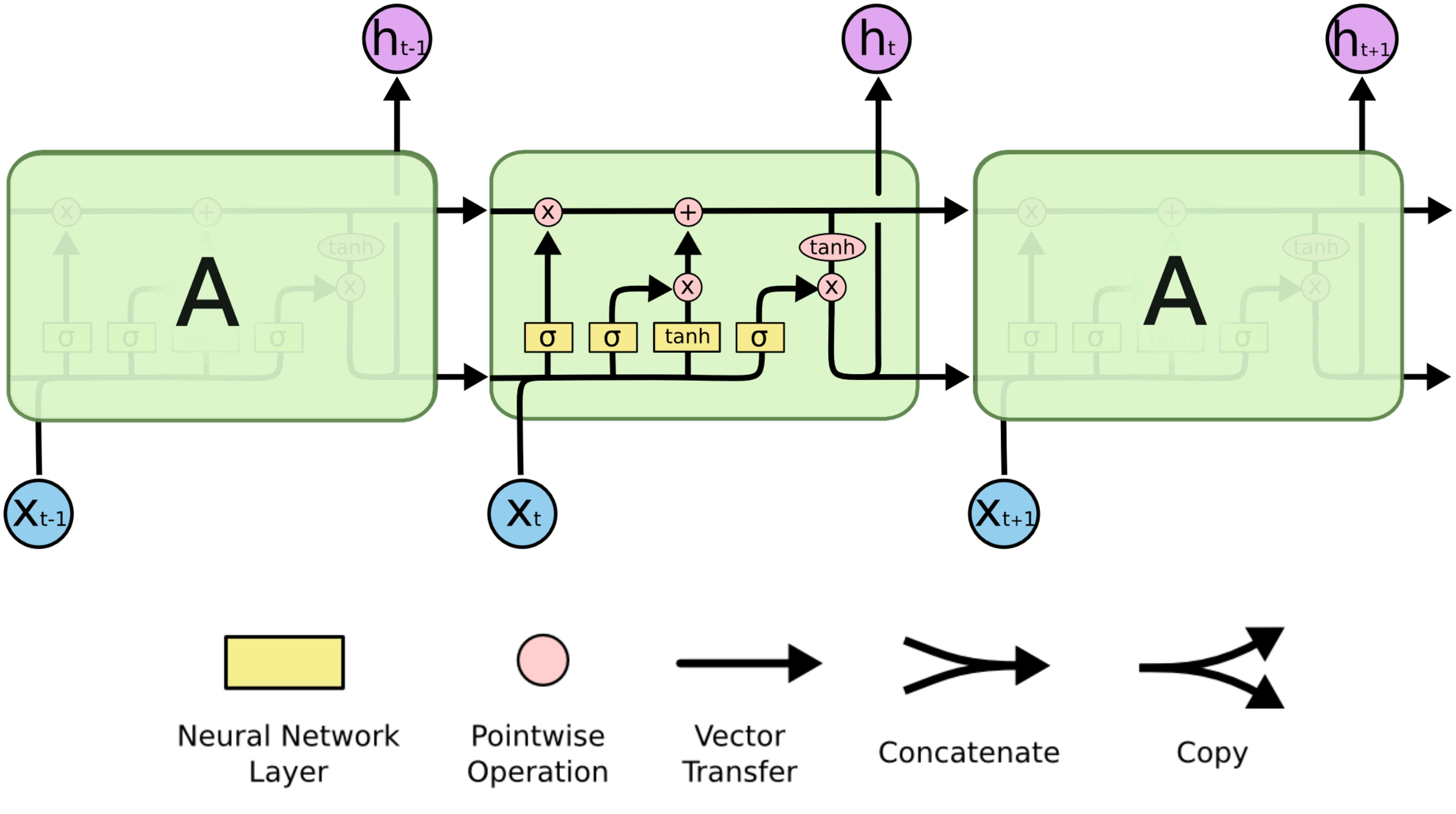
Deeper Bidirectional



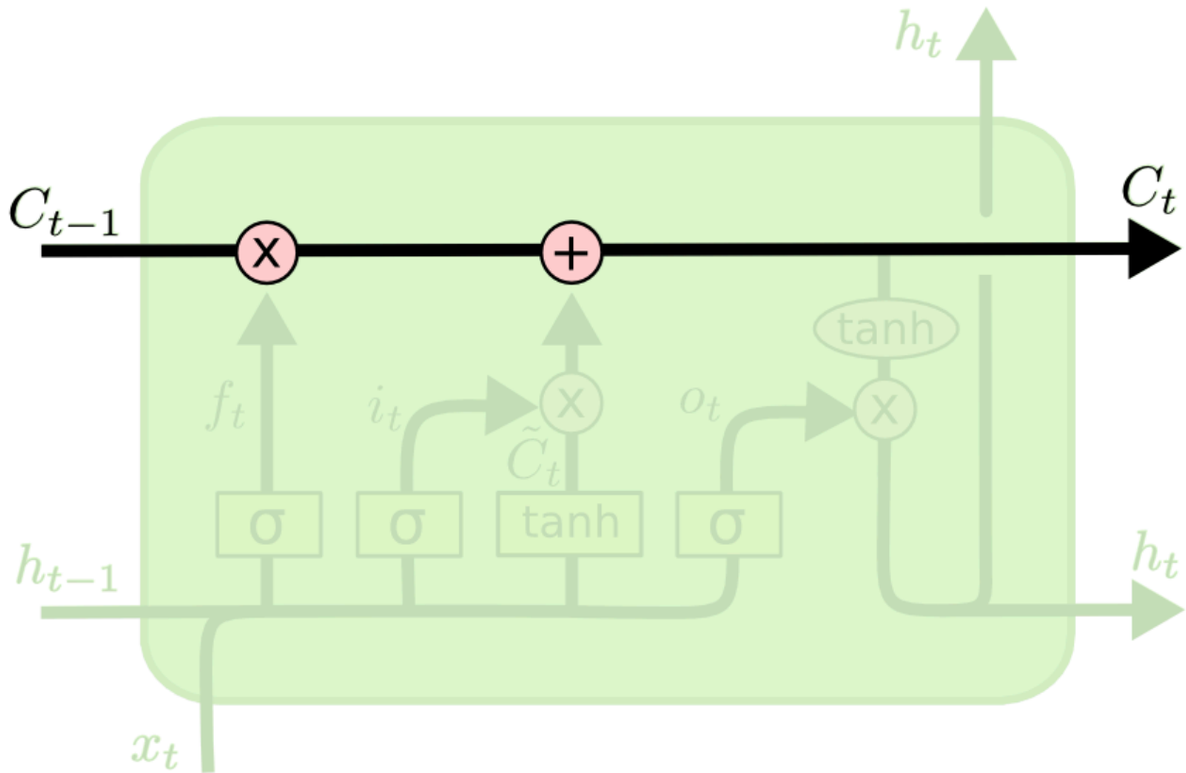
Bidirectional

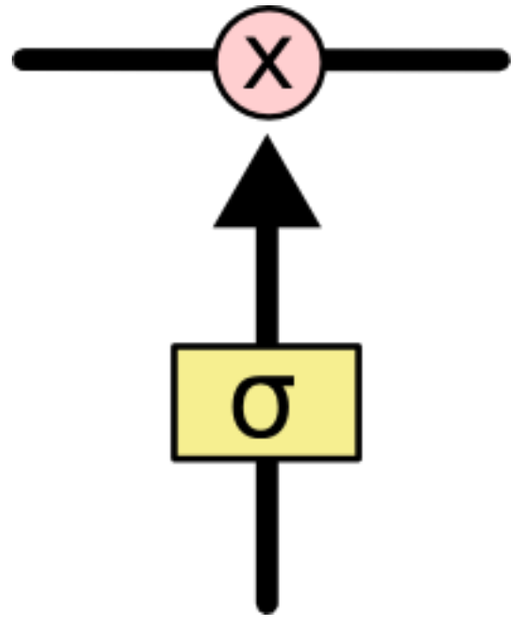
Source	<START>	The	poor	don’t	have	any	money	<END>
Forward	<START>	The	poor	don’t	have	any	money	<END>
Backward	<END>	money	any	have	don’t	poor	The	<START>

Long Short Term Memories

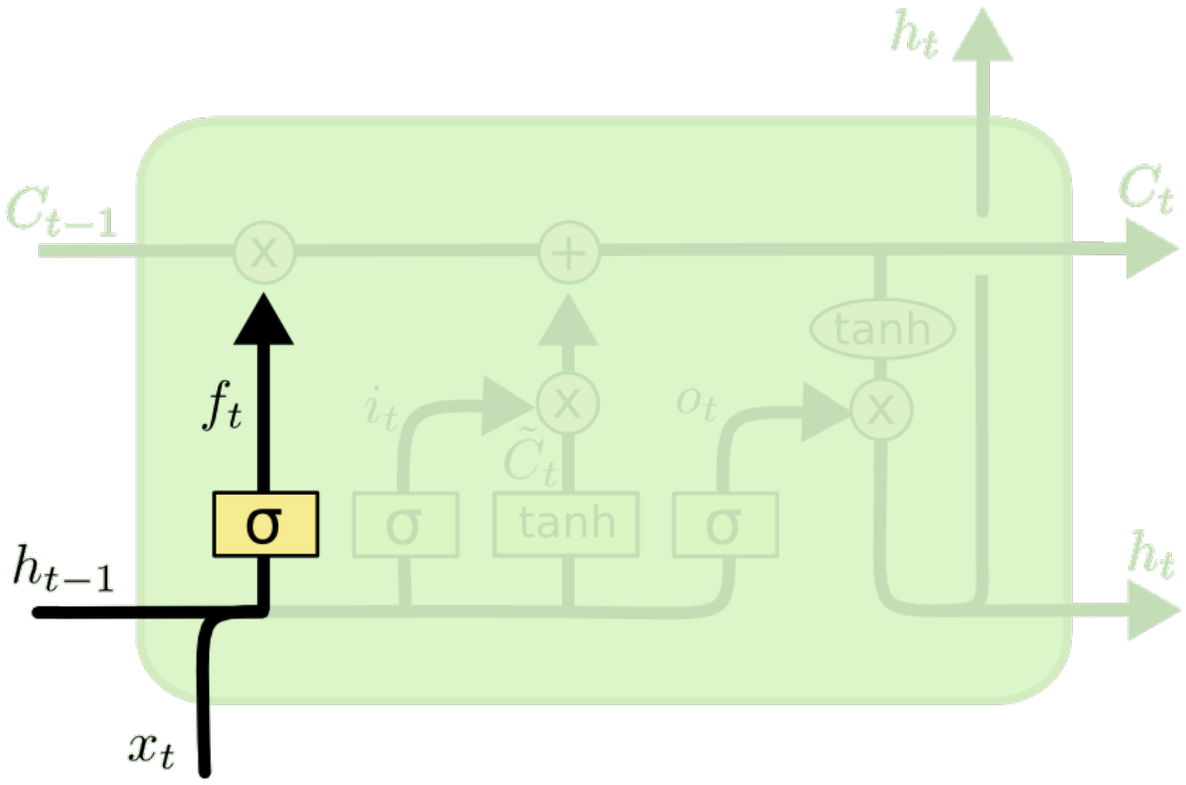


Long Short Term Memories



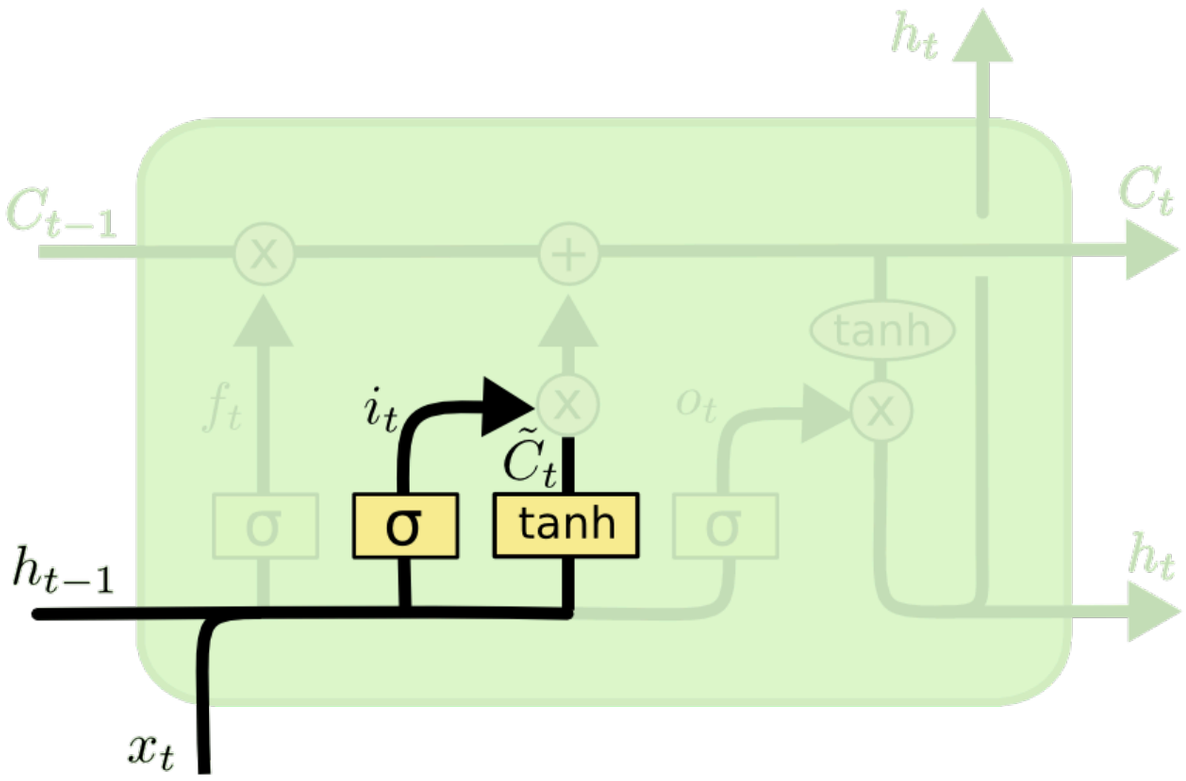


Long Short Term Memories



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

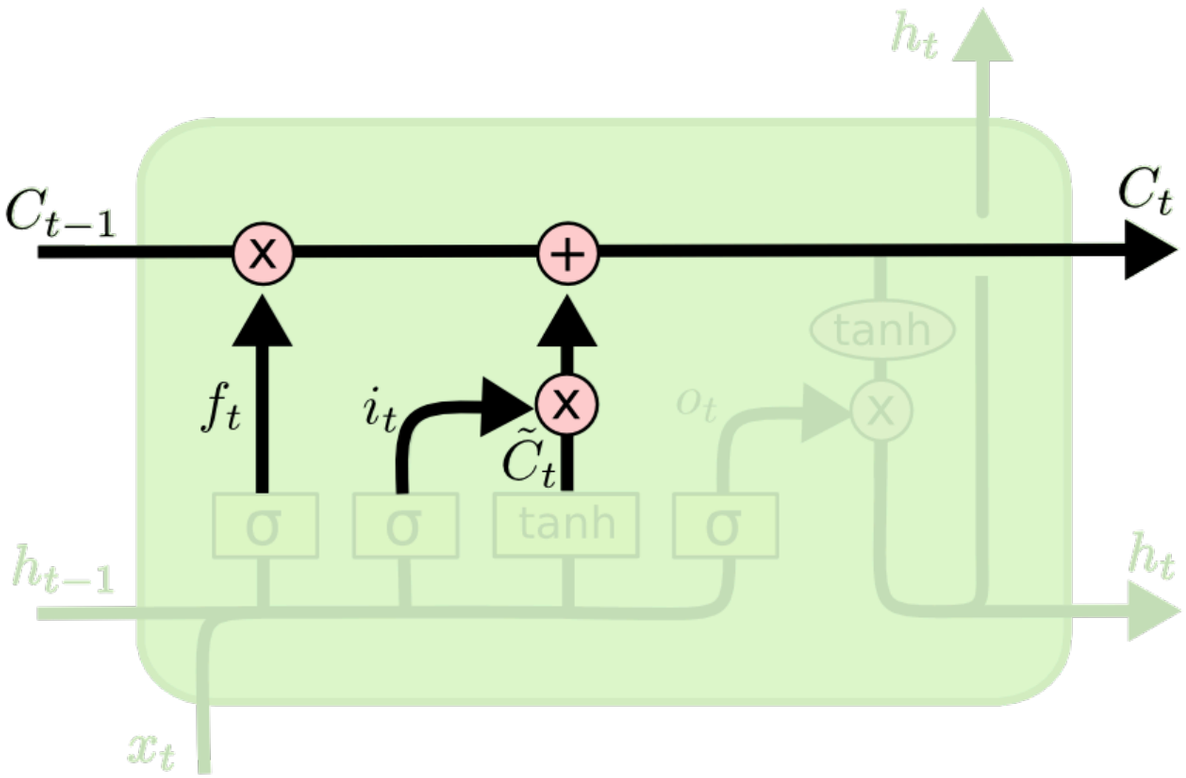
Long Short Term Memories



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

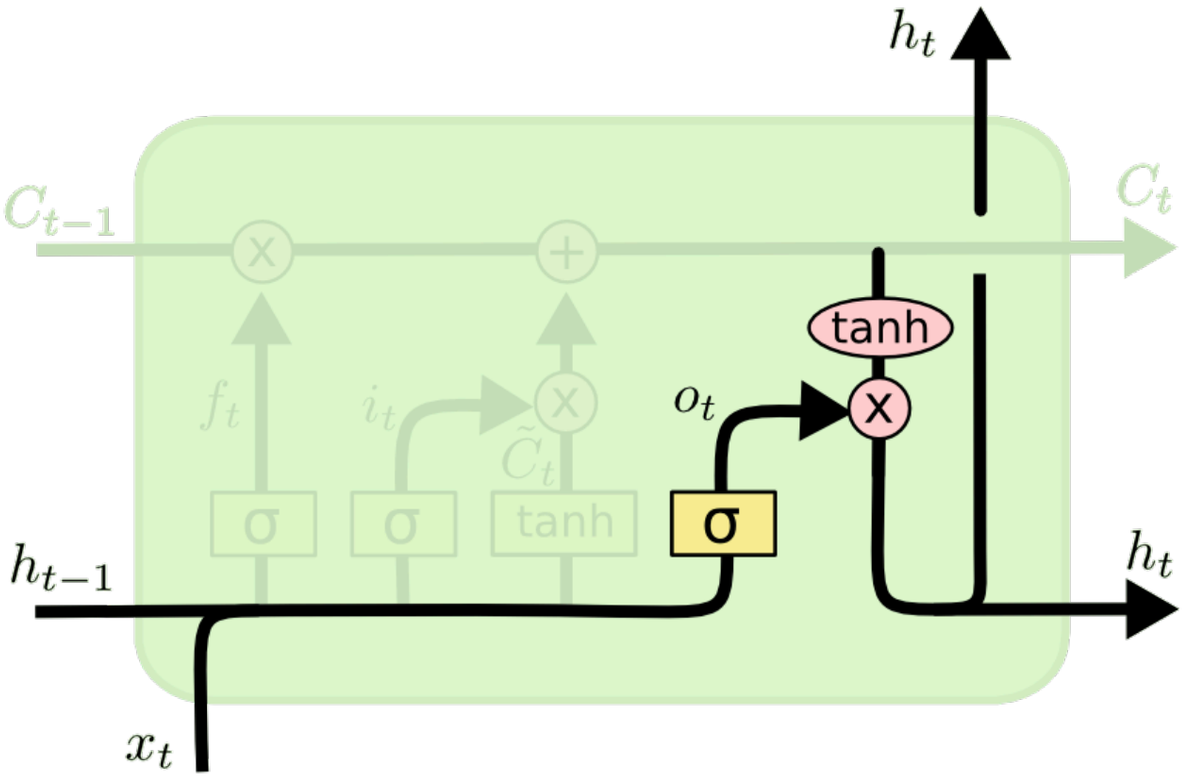
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Long Short Term Memories



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

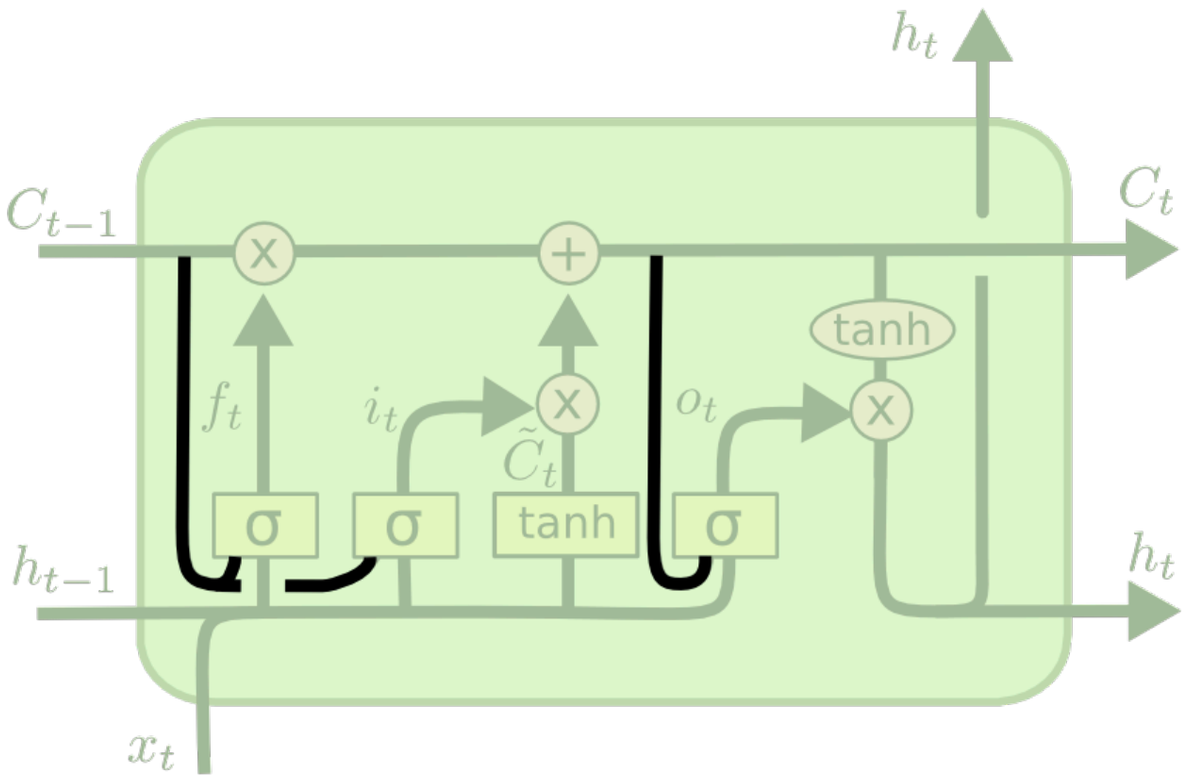
Long Short Term Memories



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

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

Long Short Term Memories

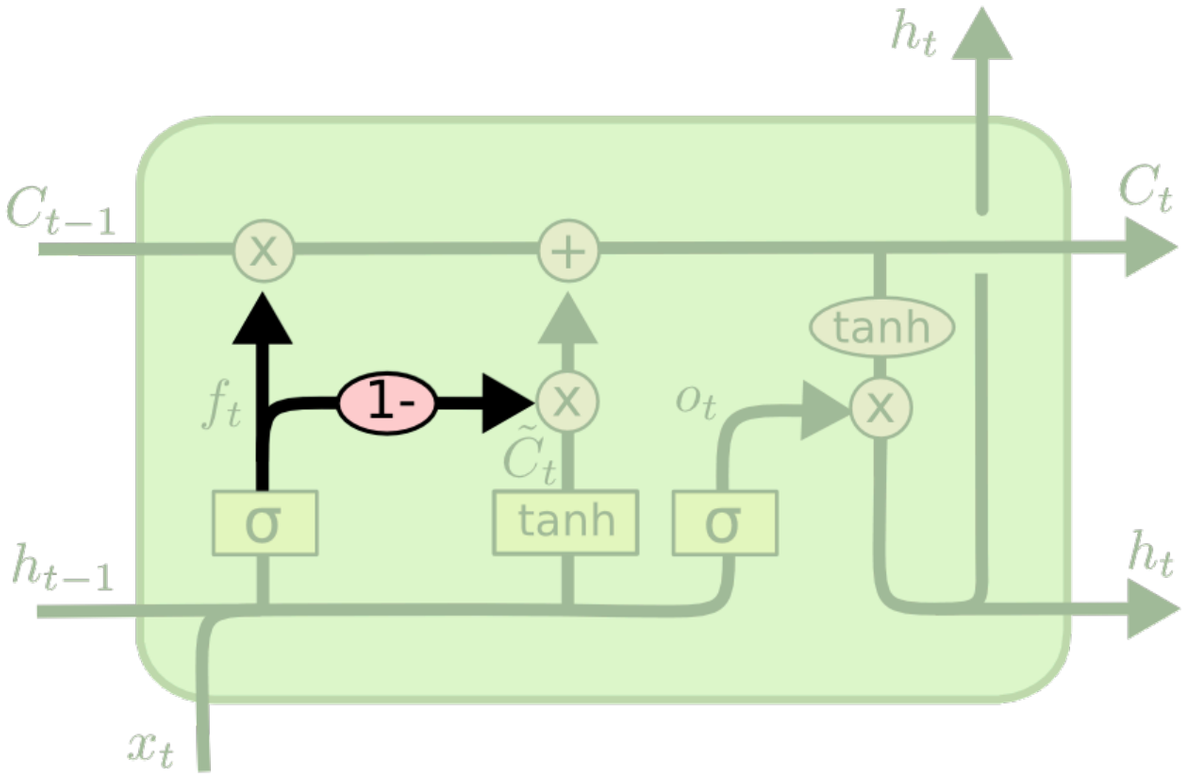


$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_f\right)$$

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

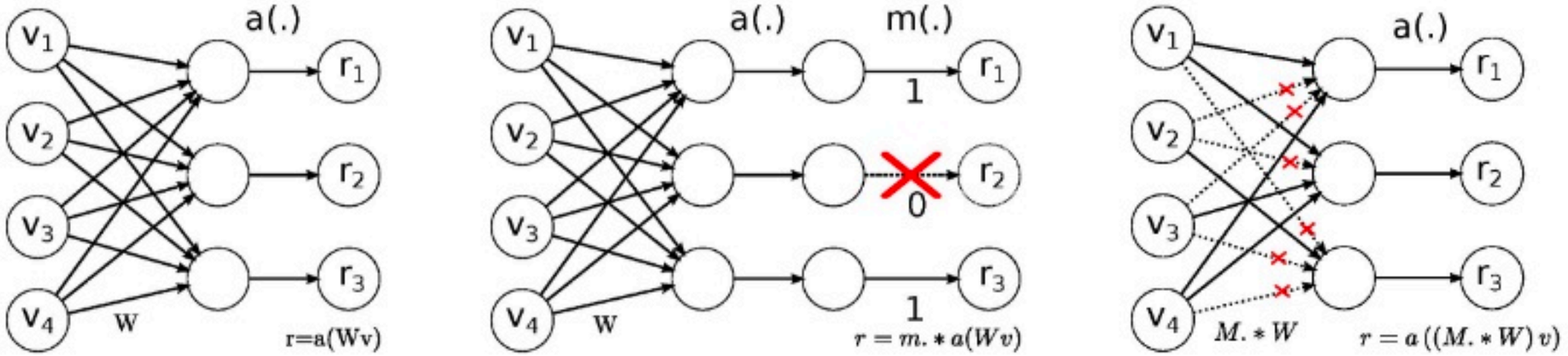
$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] \; + \; b_o\right)$$

Long Short Term Memories



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

Dropout in RNN



No-Drop Network

DropOut Network

DropConnect Network

Layer Normalization

Batch Normalization → **Layer Normalization**

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

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