AMRITA
VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

Convolutional neural network (CNN)
&
Recurrent neural network (RNN)

Vinayakumar R

PhD Student,

Centre for Computational Engineering and Networking,
Amrita Vishwa Vidyapeetham,

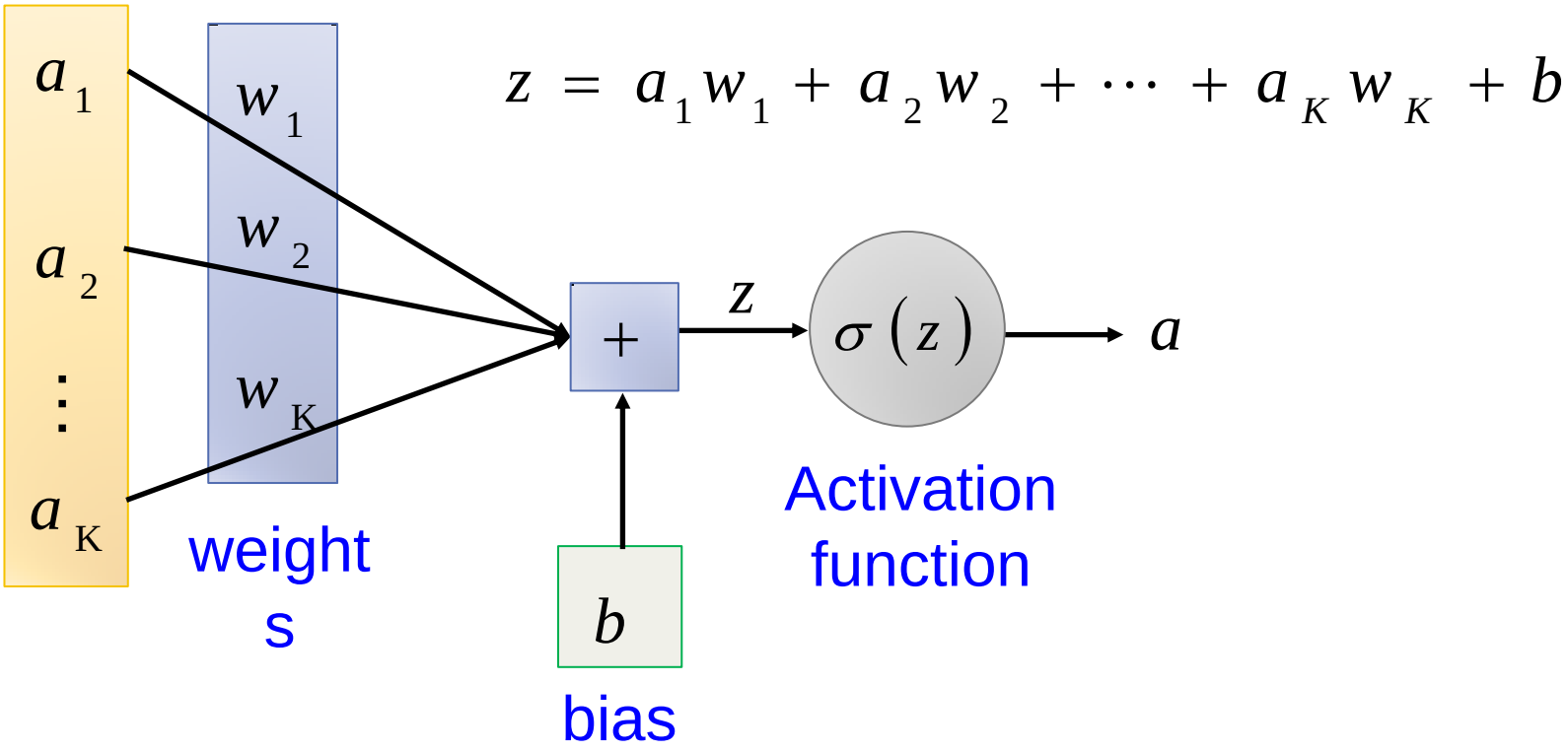
Coimbatore

https://vinayakumarr.github.io/

AMRITA VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

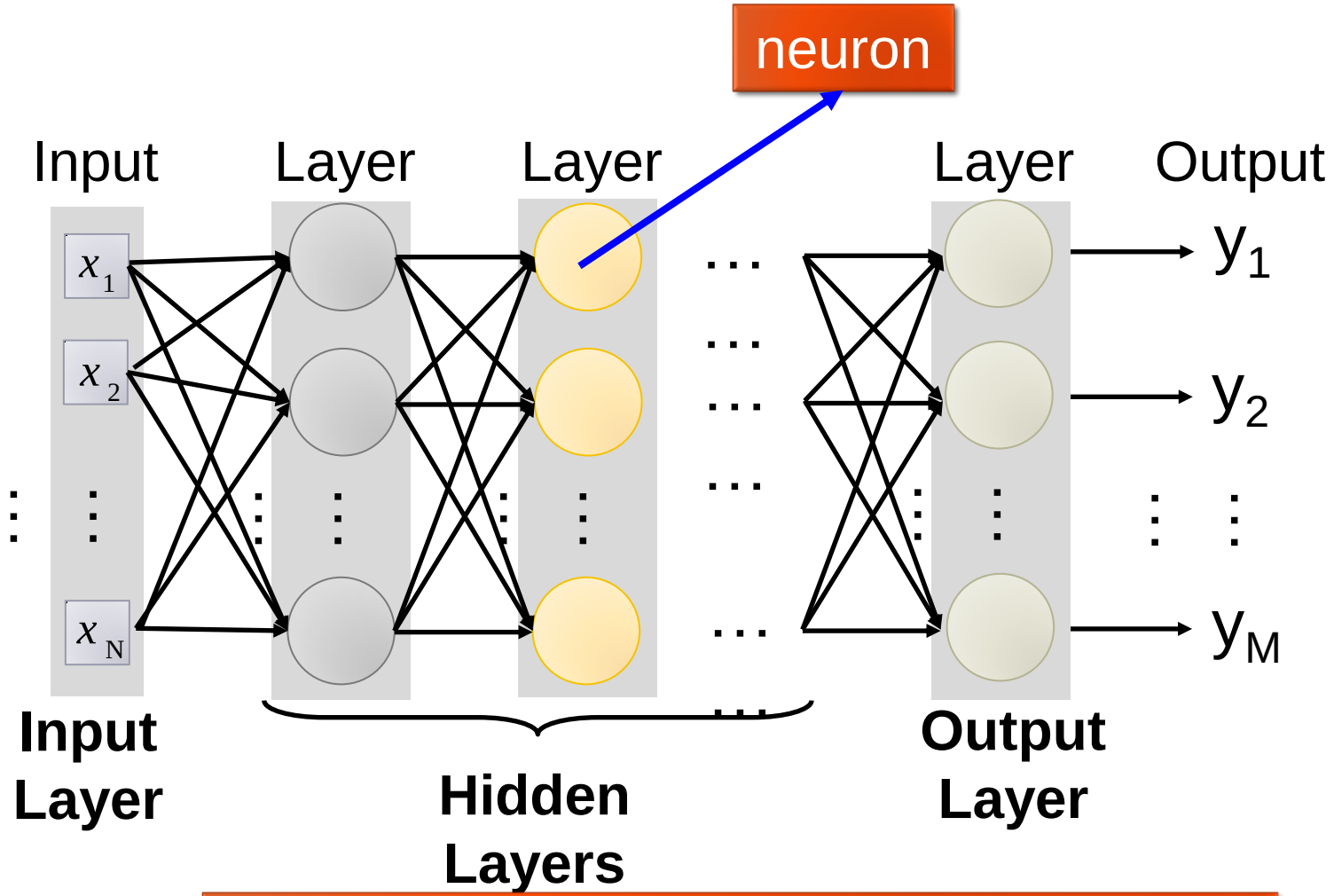
Element of Neural Network

Neuron $f: R^K \rightarrow R$



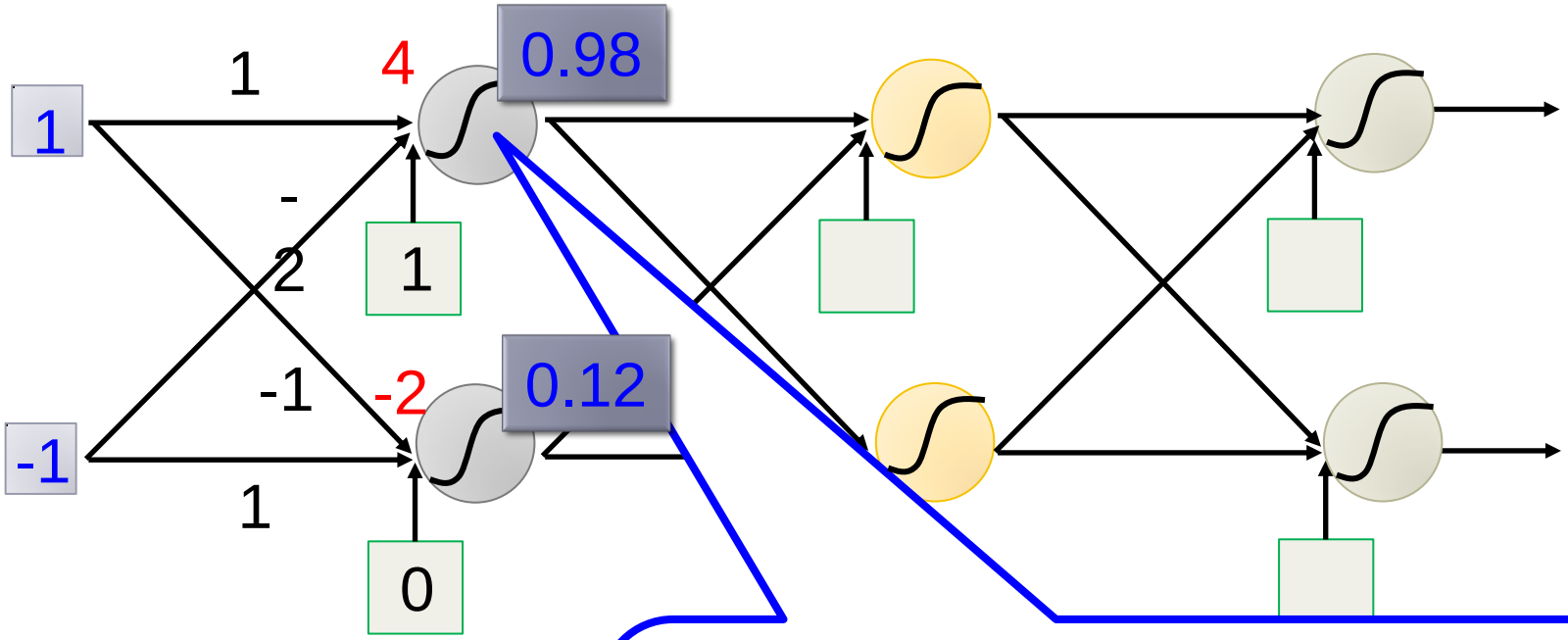
$$z = a_1 w_1 + a_2 w_2 + \cdots + a_K w_K + b$$

Neural Network



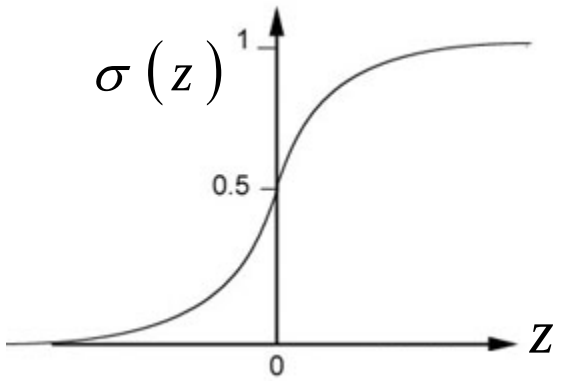
Deep means many hidden layers

Neural Network

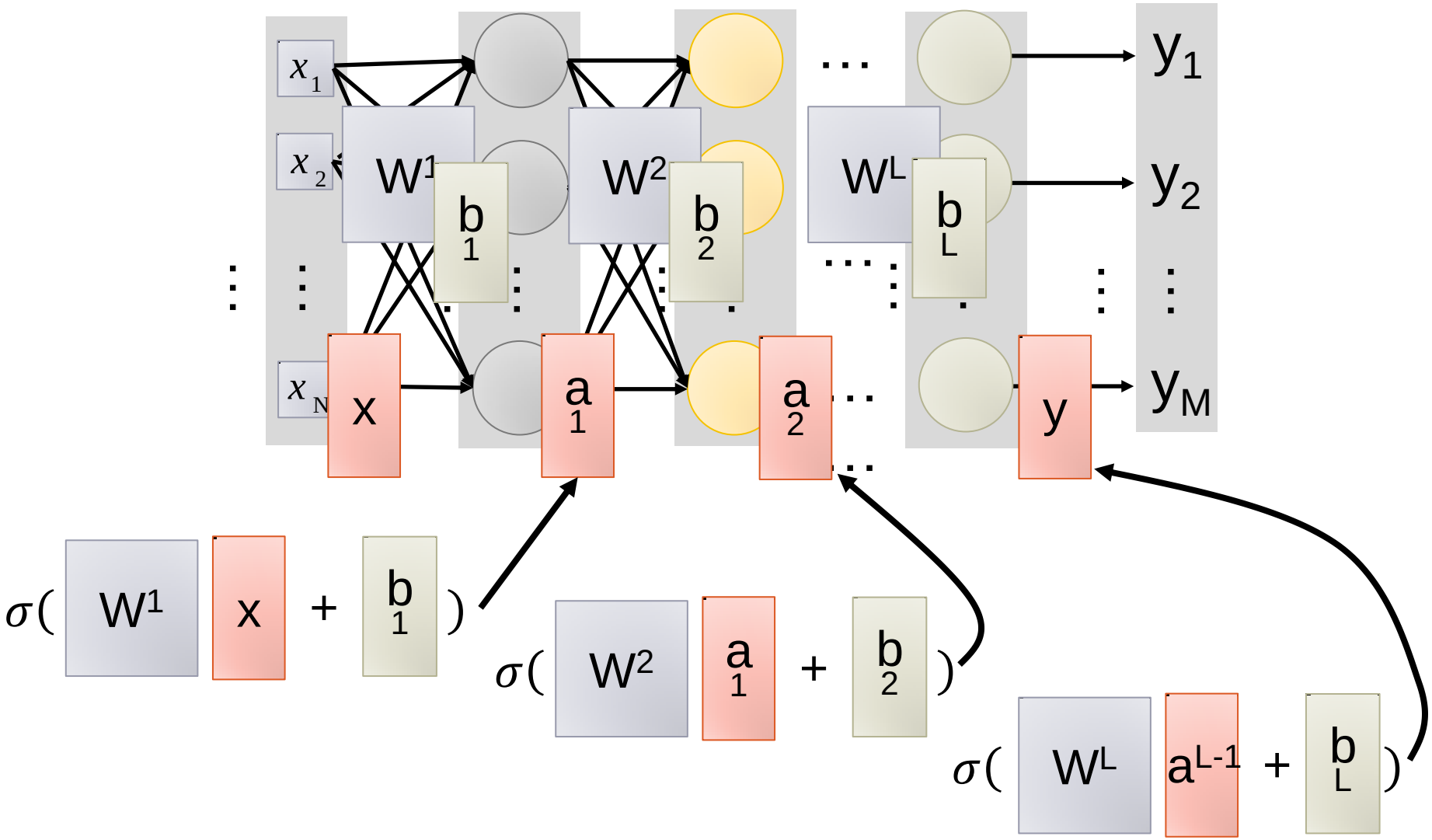


Sigmoid Function

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$



Neural Network

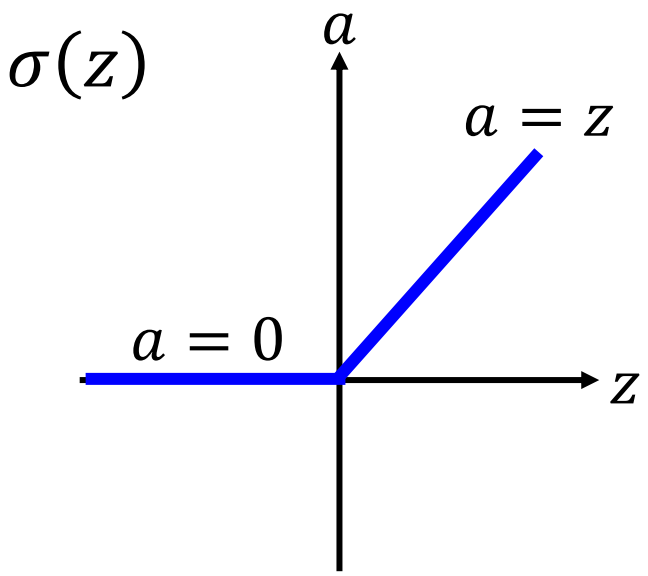


Training DNN

New Activation Function

ReLU

- Rectified Linear Unit (ReLU)



Reason:

1. Fast to compute

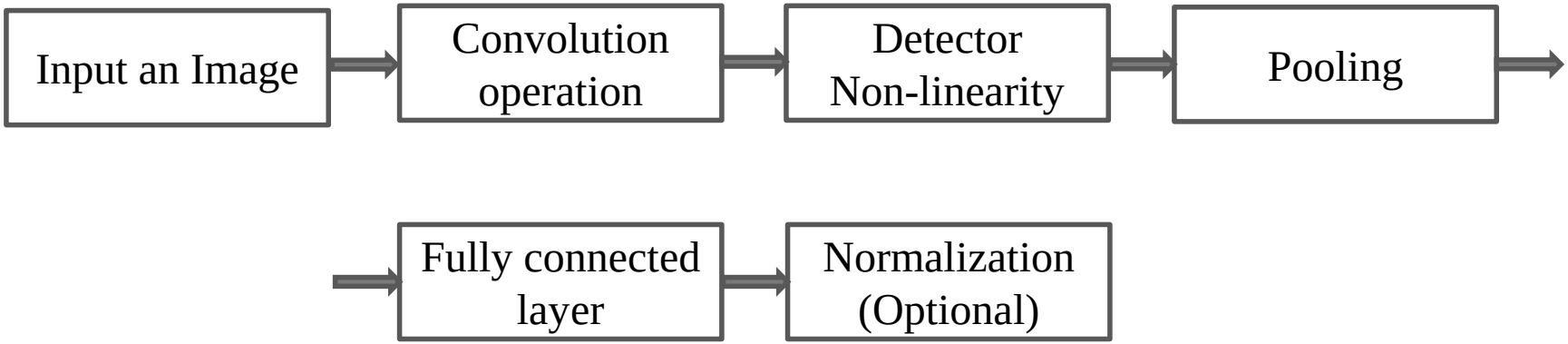
2. Vanishing gradient problem

$$f'(x) = \begin{cases} 1 \; if \; x >= 0 \\ 0 \; if \; x < 0 \end{cases}$$

$$\sigma'(z) = \sigma(z) * (1 - \sigma(z))$$

Convolutional Neural Network

- Neural Network with a convolution operation instead of matrix multiplication in at least oneof the layers



Convolutional Neural Network

AMRITA
VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

Input, e.g. an image

1	3	5	2	4
6	0	2	1	3
6	3	1	3	6
7	3	2	1	3
5	3	0	0	2

Filter (Kernel)

0.2	0.7
-0.5	0.7

Convolutional Neural Network

- Input, e.g. an image

1	3	5	2	4
6	0	2	1	3
6	3	1	3	6
7	3	2	1	3
5	3	0	0	2

$$c_1 = \mathrm{f}\,(0.2 * 1 + 0.7 * 3 - 0.5 * 6 + 0.7 * 0) = \mathrm{f}(-0.7)$$

$$c_2 = \mathrm{f}(0.2 * 3 + 0.7 * 5 - 0.5 * 0 + 0.7 * 2) = \mathrm{f}(5.5)$$

Note: Bias terms omitted!

f represents some non-linear activation function

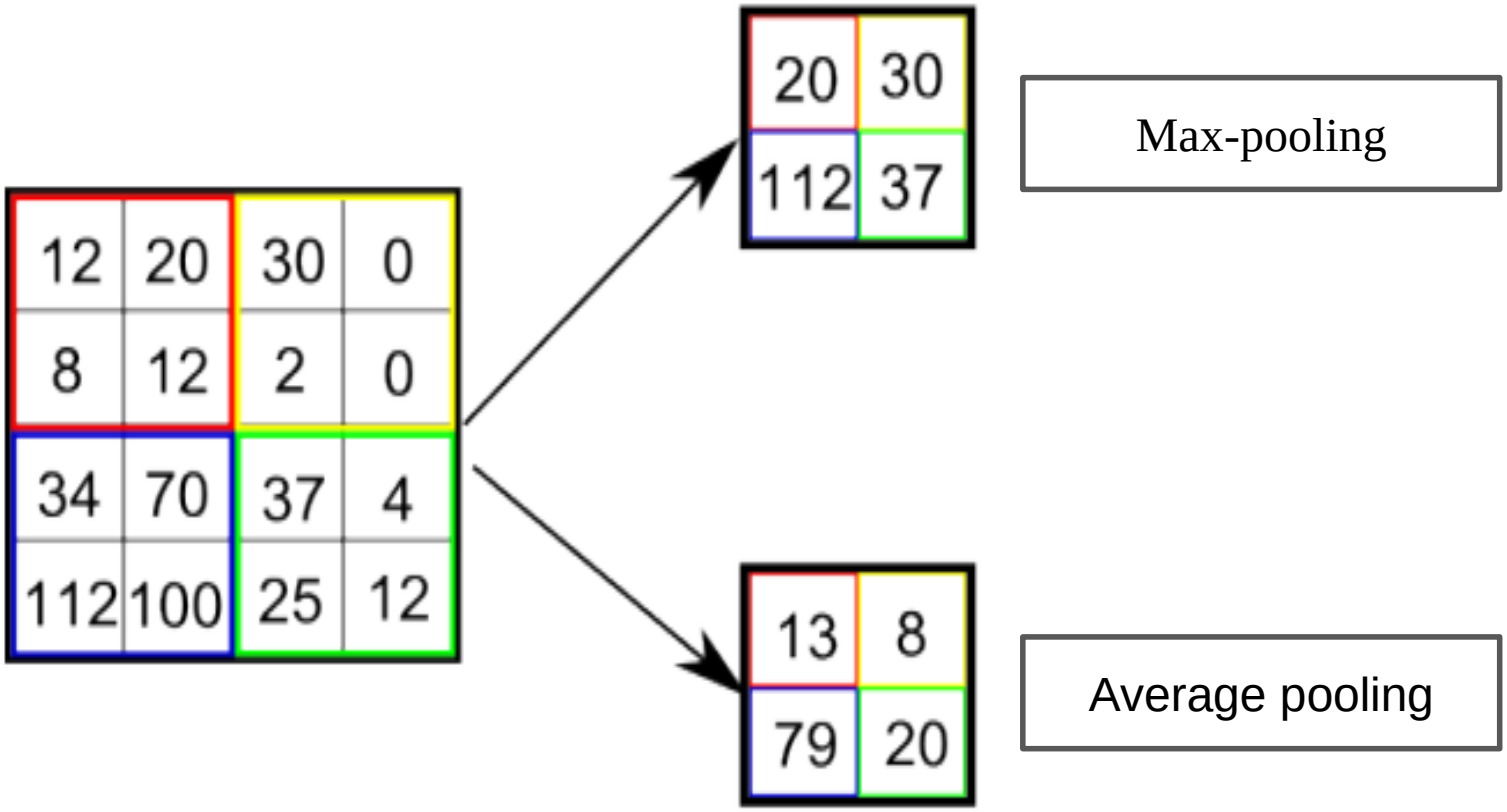
Filter (Kernel)

0.2	0.7
-0.5	0.7

Feature map

f(-0.7) f(5.5) …

Convolutional Neural Network



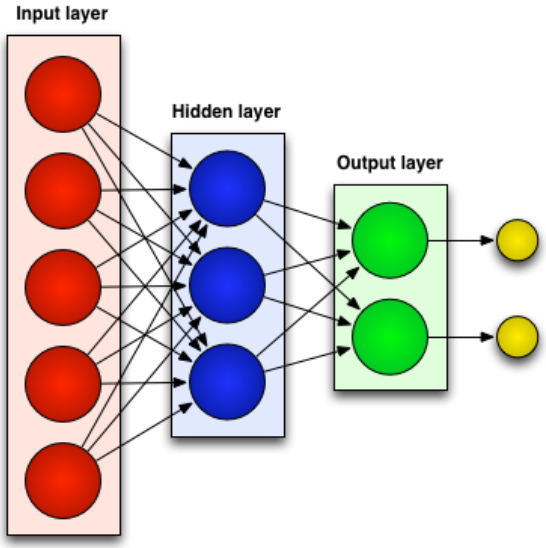
Recurrent Neural Networks

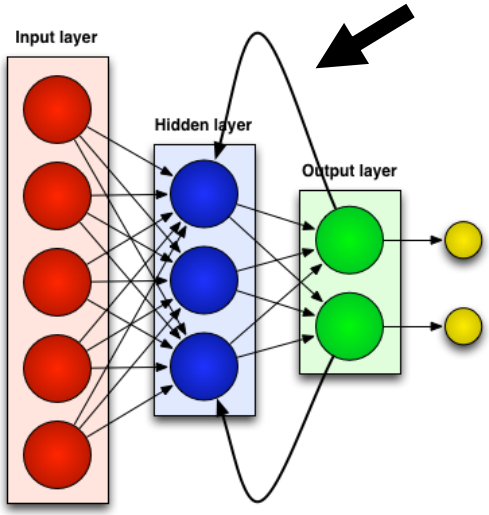
AMRITA VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

- **Generally there are two kinds of neural networks:**

➢ **Feedforward Neural Networks:**

✓ **connections between the units do not form a cycle**



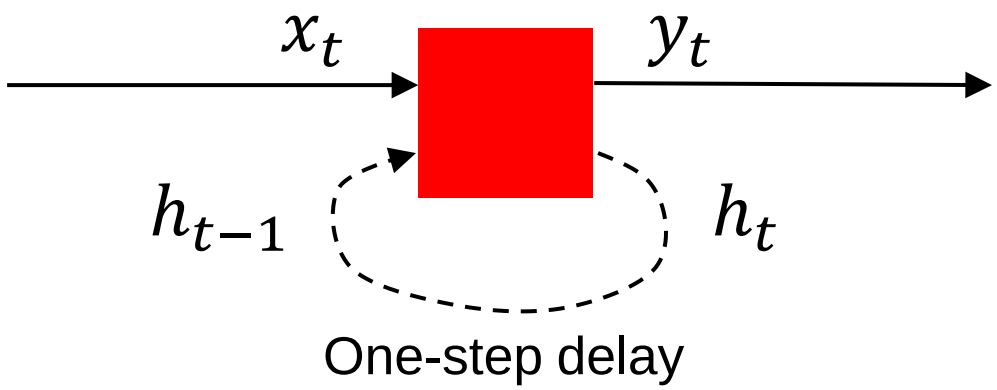


➢ **Recurrent Neural Network:**

✓ **connections between units form cyclic paths**

Recurrent Neural Networks

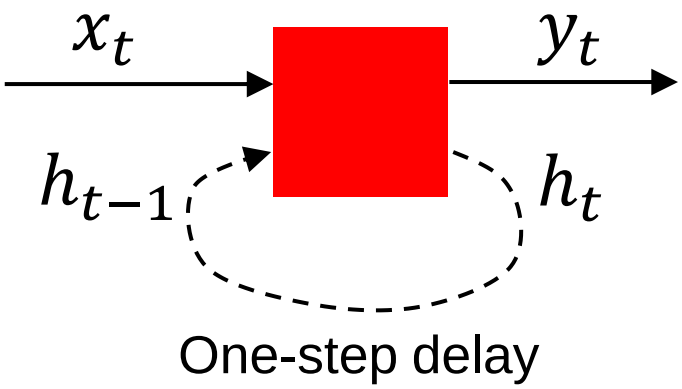
Recurrent networks introduce cycles and a notion of time.

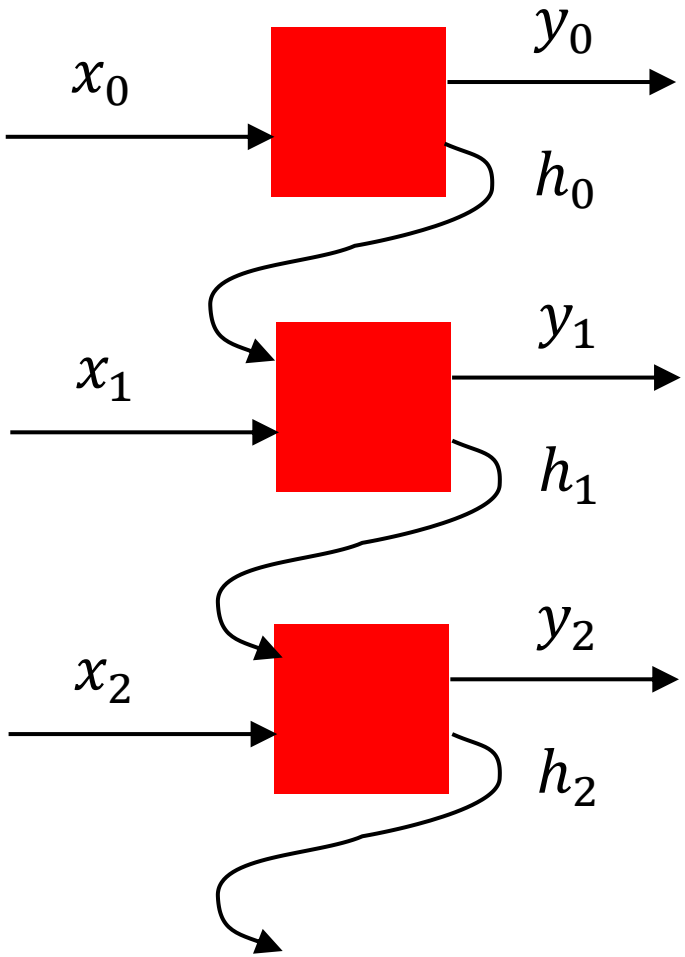


- They are designed to process sequences of data $x_1, \ldots, x_n$ and can produce sequences of outputs $y_1, \ldots, y_m$.

Unrolling RNNs

RNNs can be unrolled across multiple time steps.



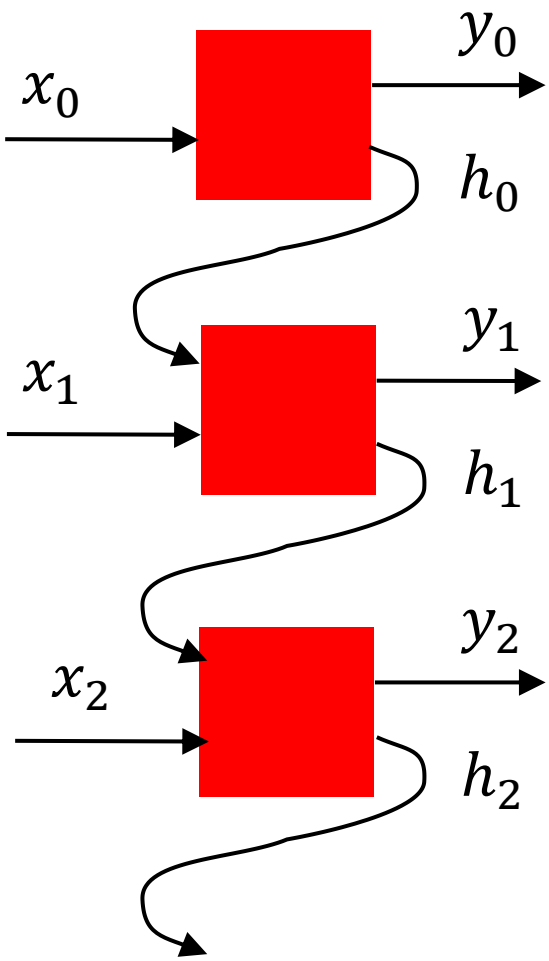


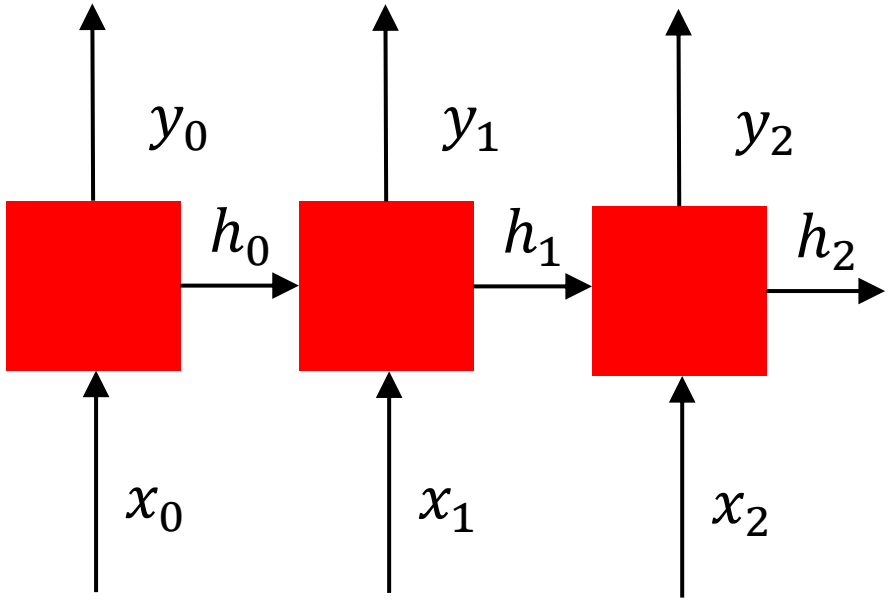
This produces a DAG which supports backpropagation.

But its size depends on the input sequence length.

Unrolling RNNs

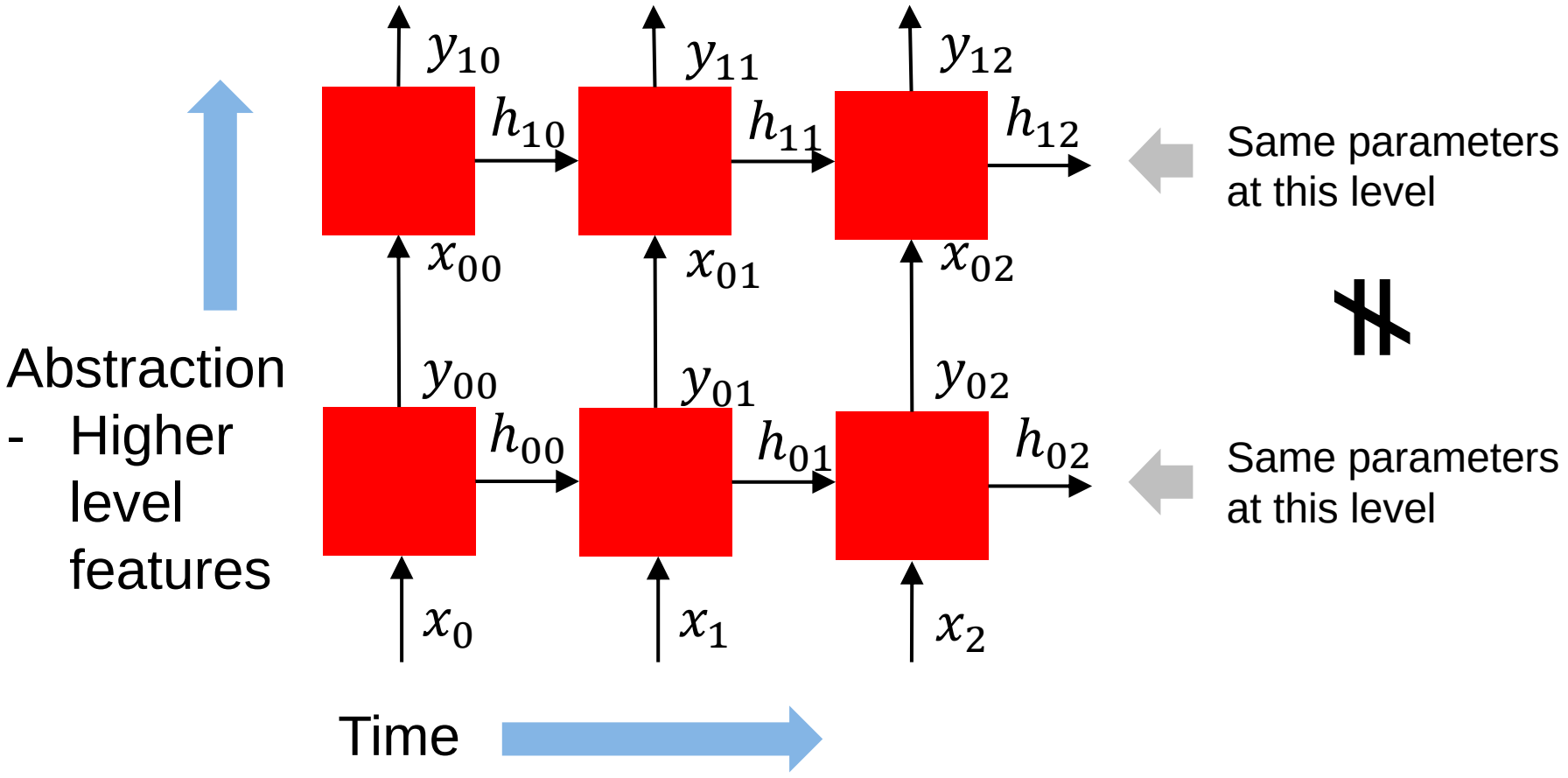
Usually drawn as:





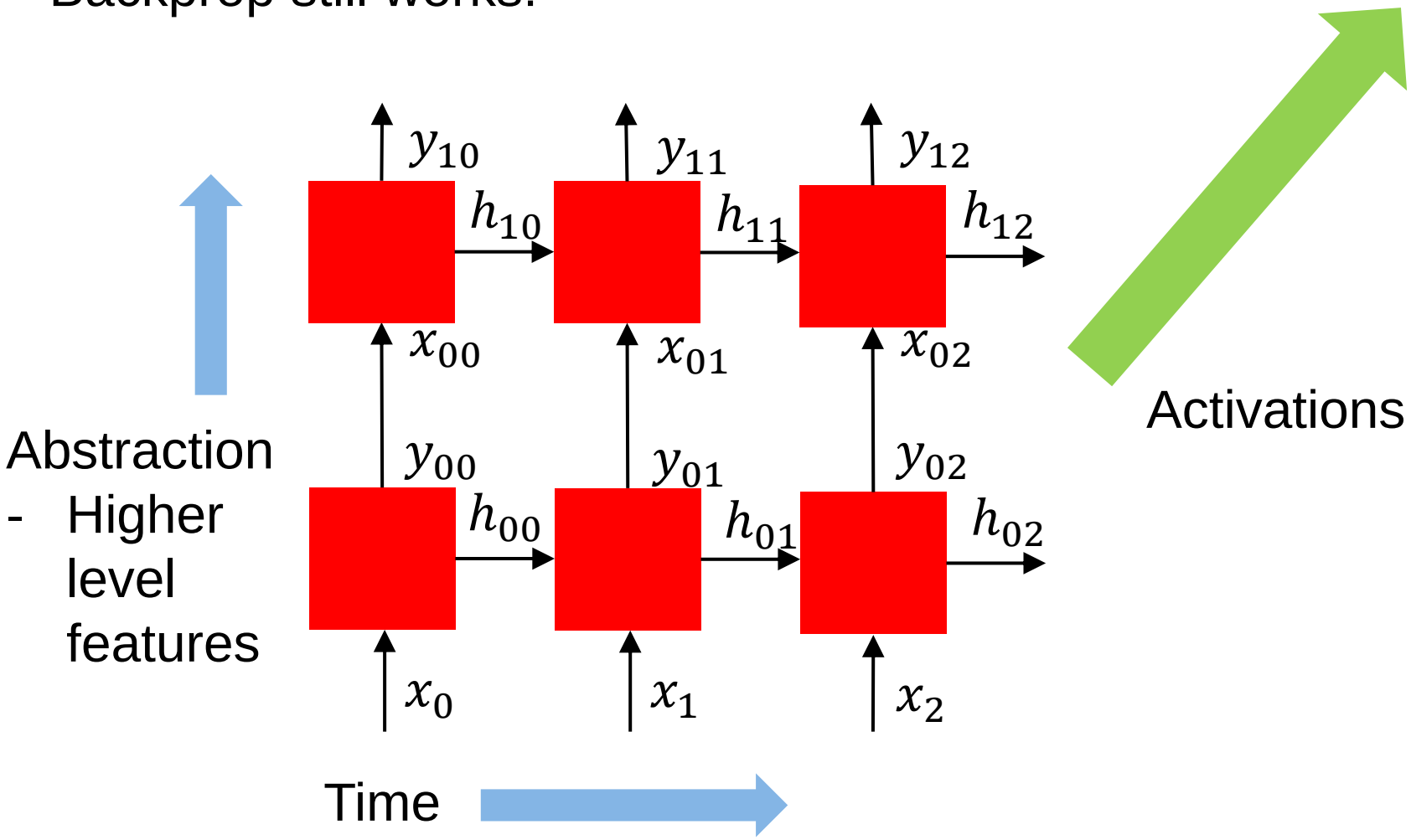
RNN structure

Often layers are stacked vertically (deep RNNs):



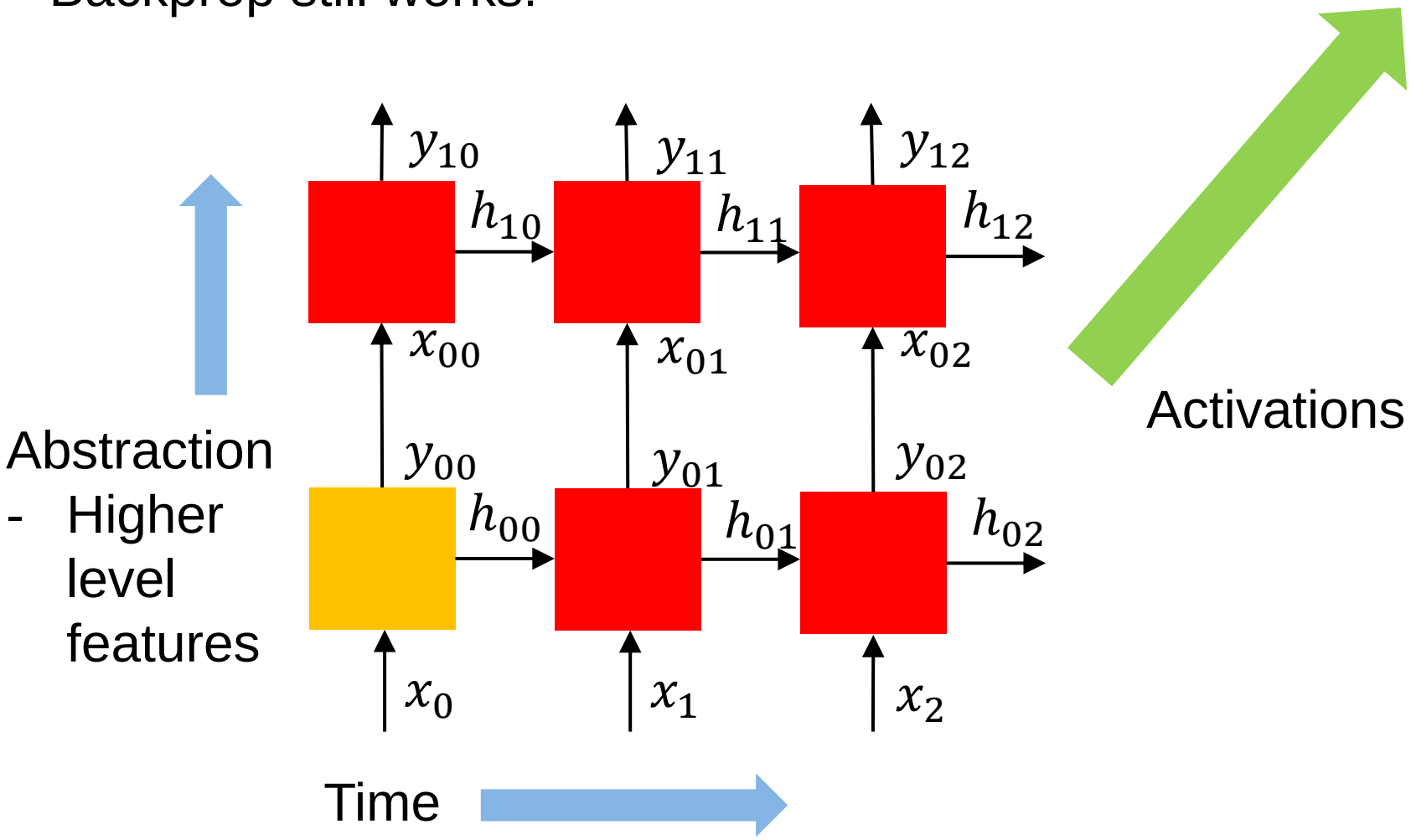
RNN structure

Backprop still works:



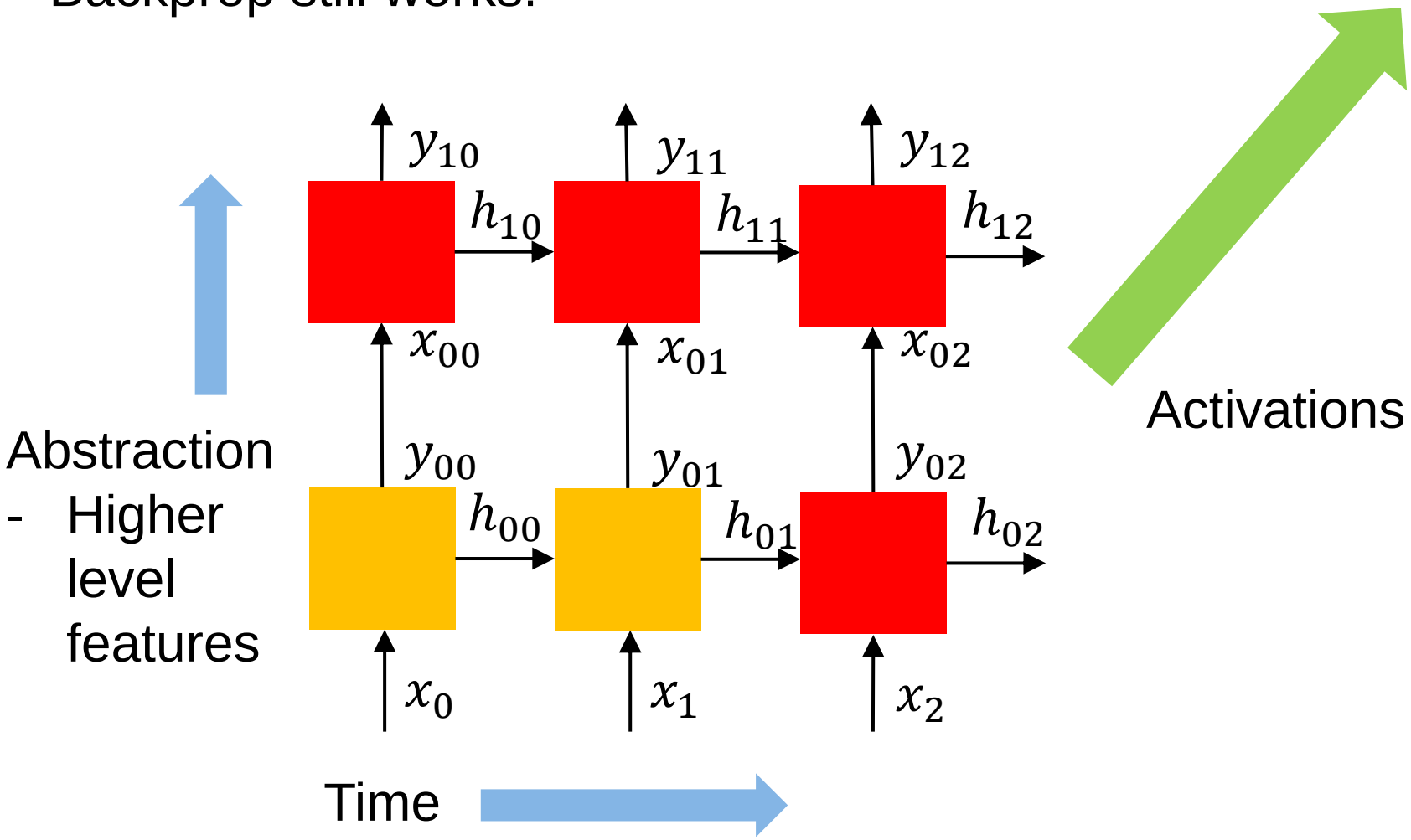
RNN structure

Backprop still works:



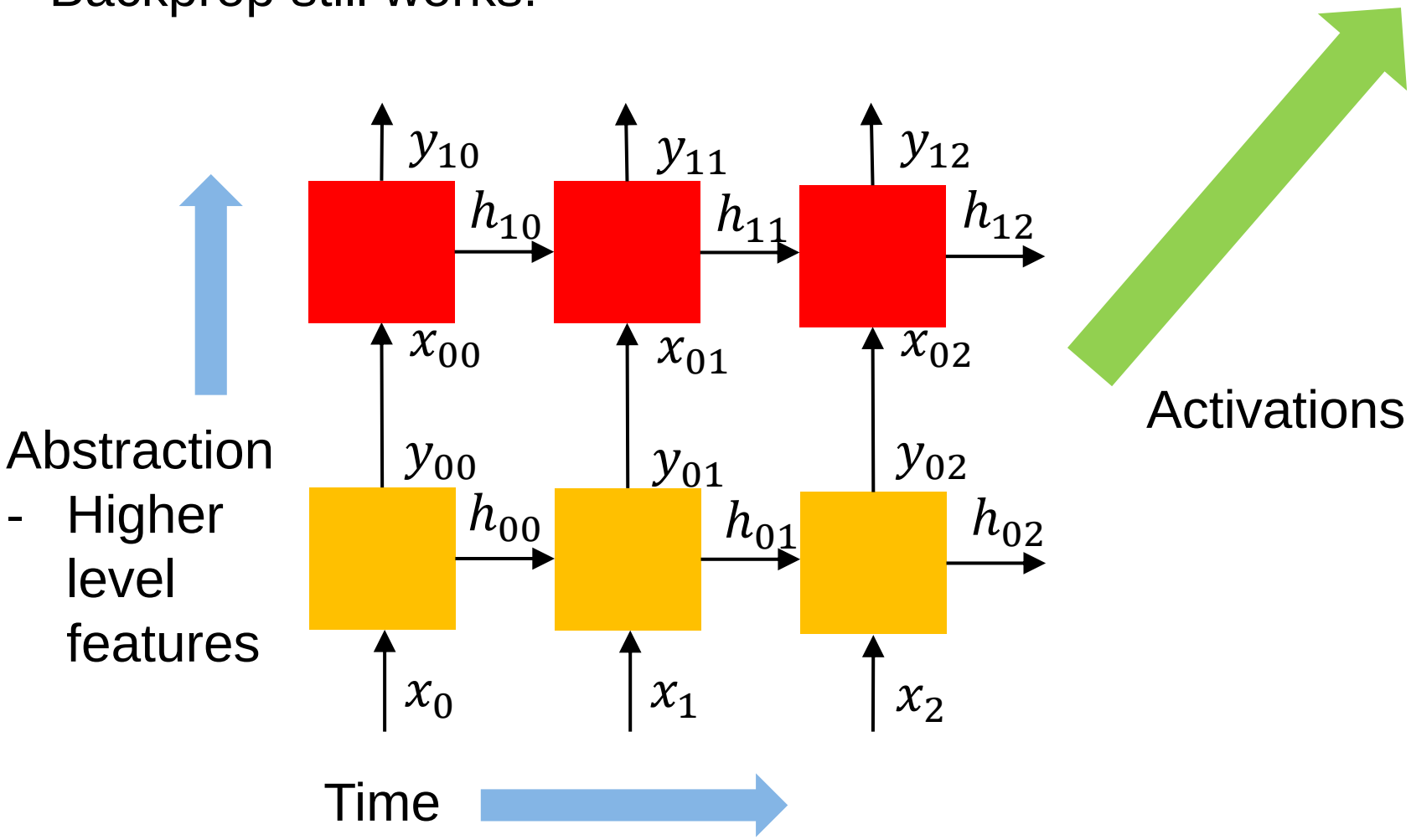
RNN structure

Backprop still works:



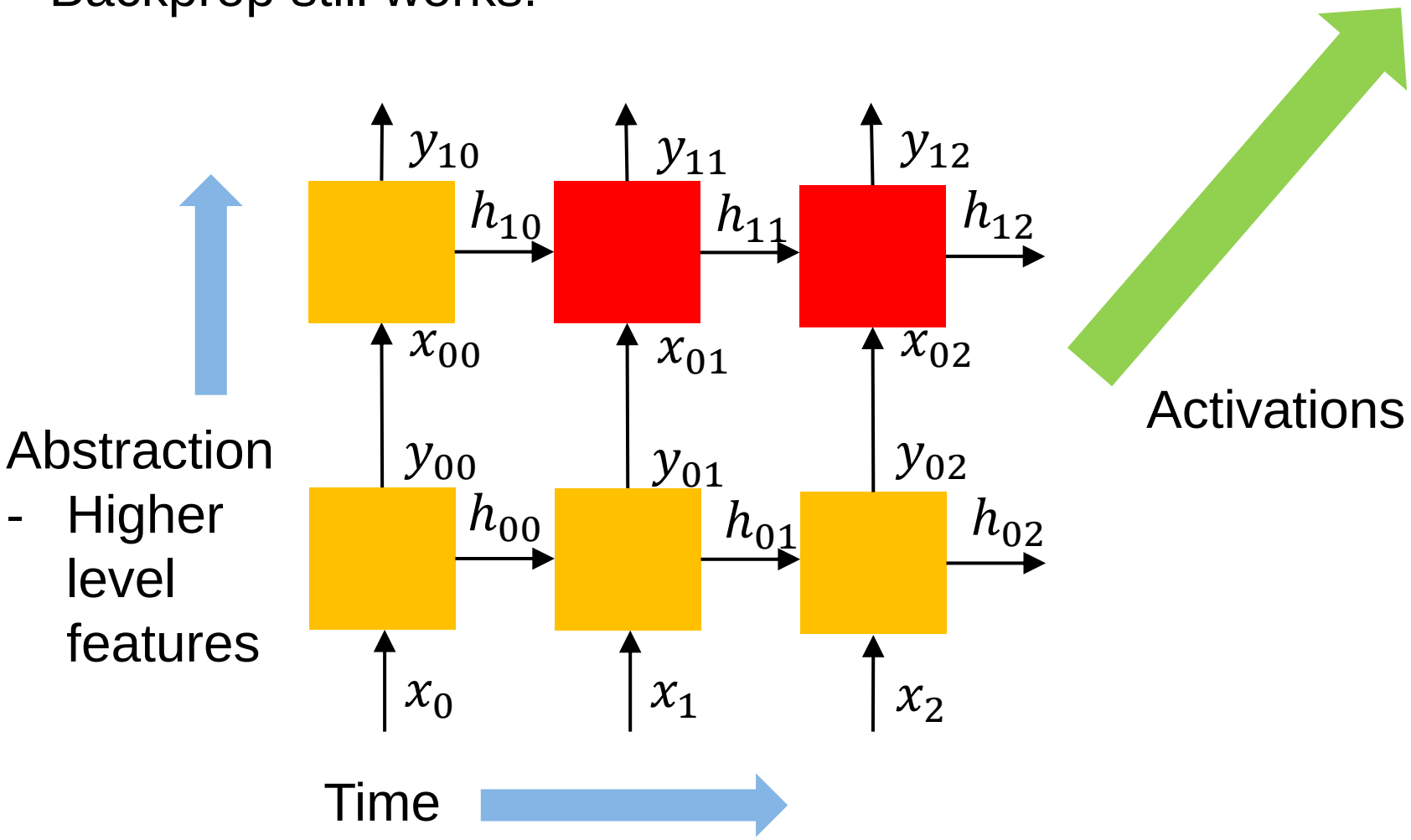
RNN structure

Backprop still works:



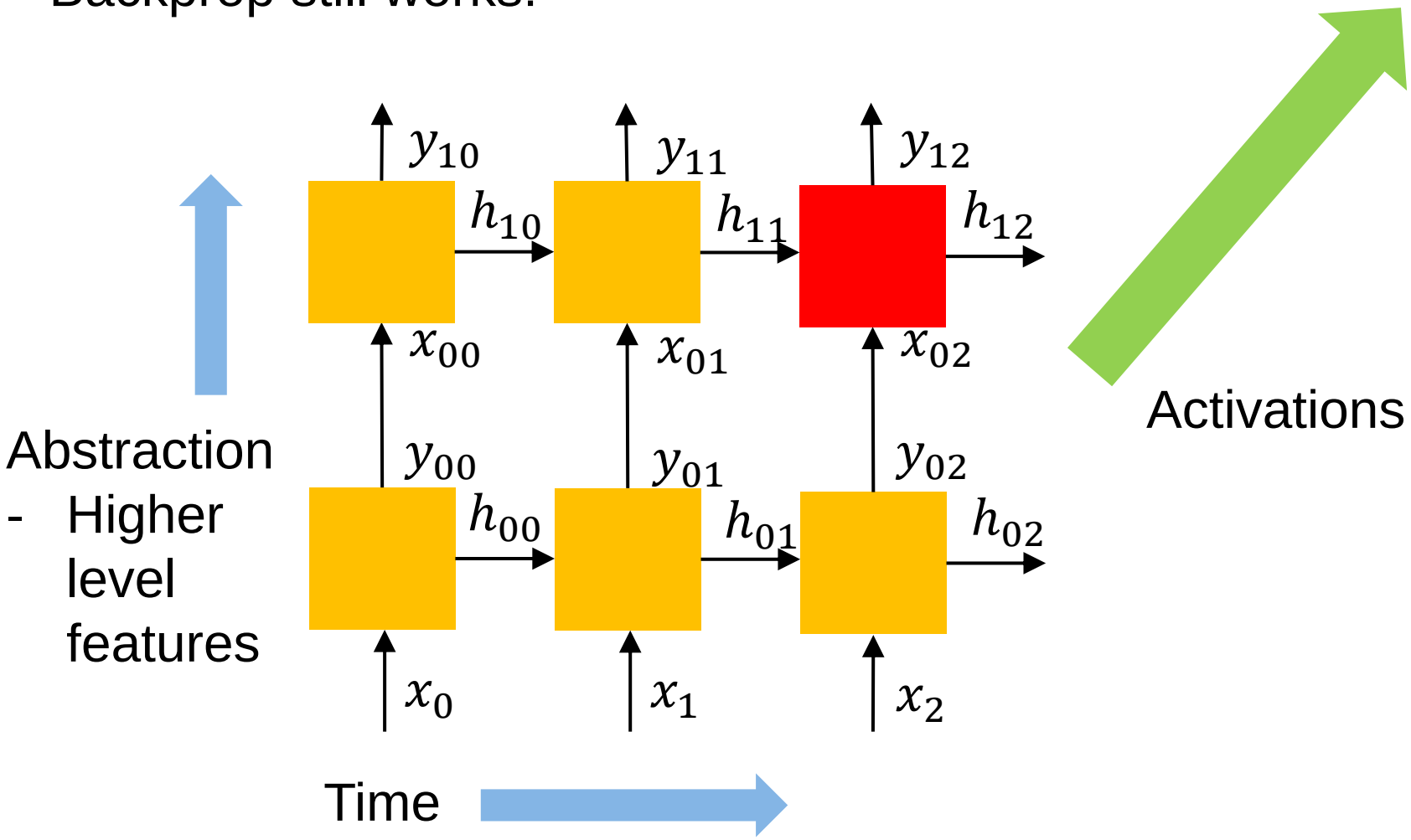
RNN structure

Backprop still works:



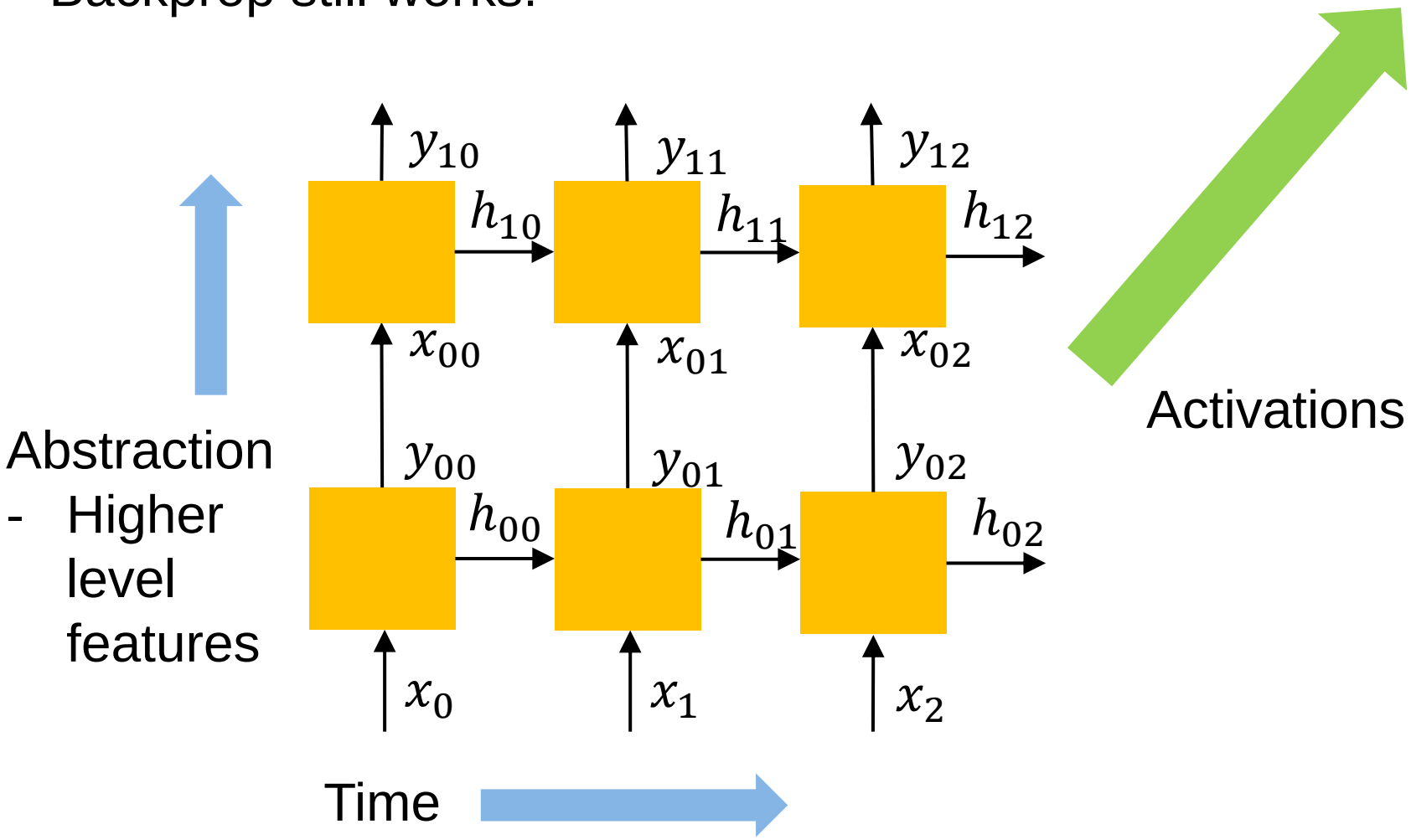
RNN structure

Backprop still works:



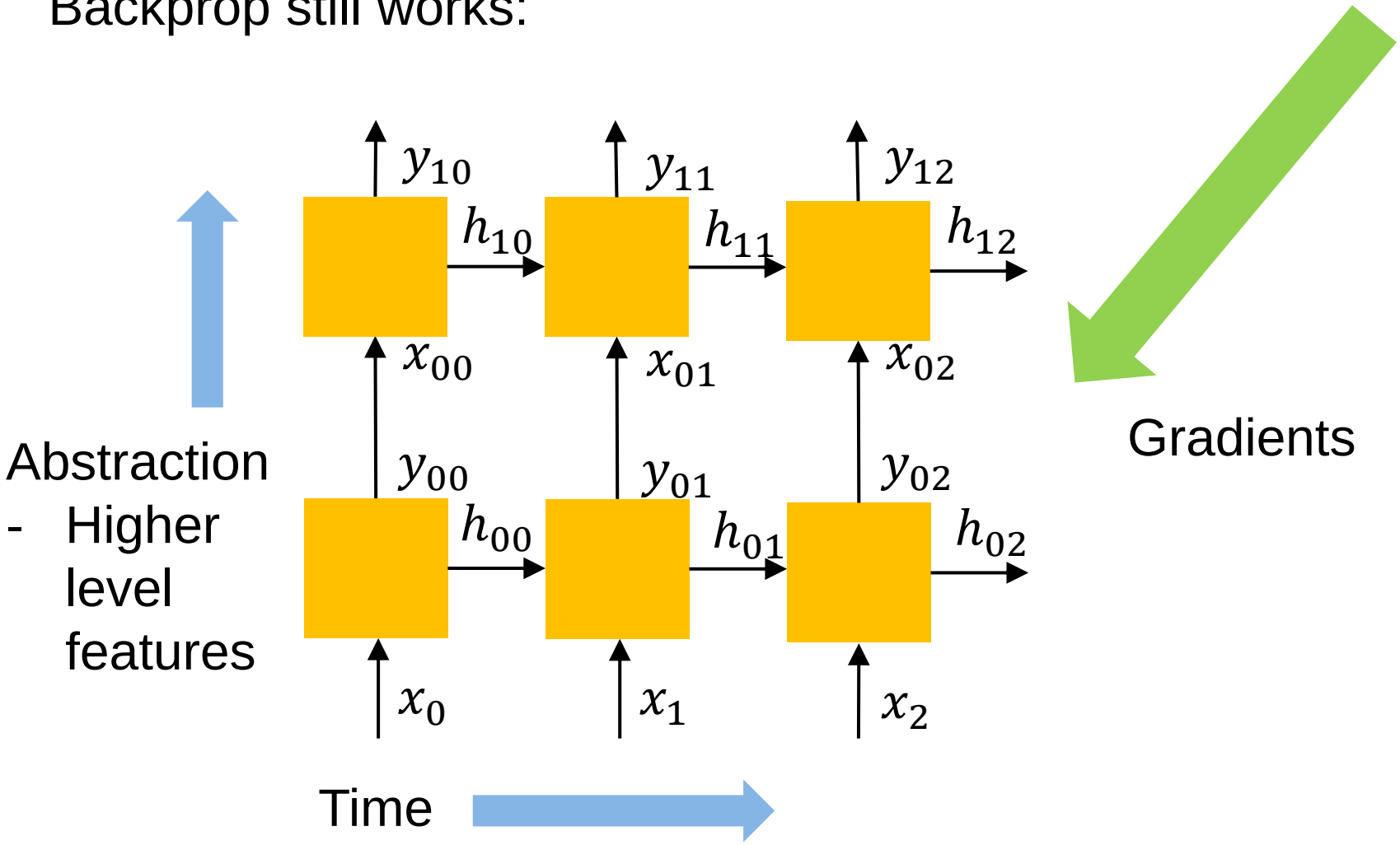
RNN structure

Backprop still works:



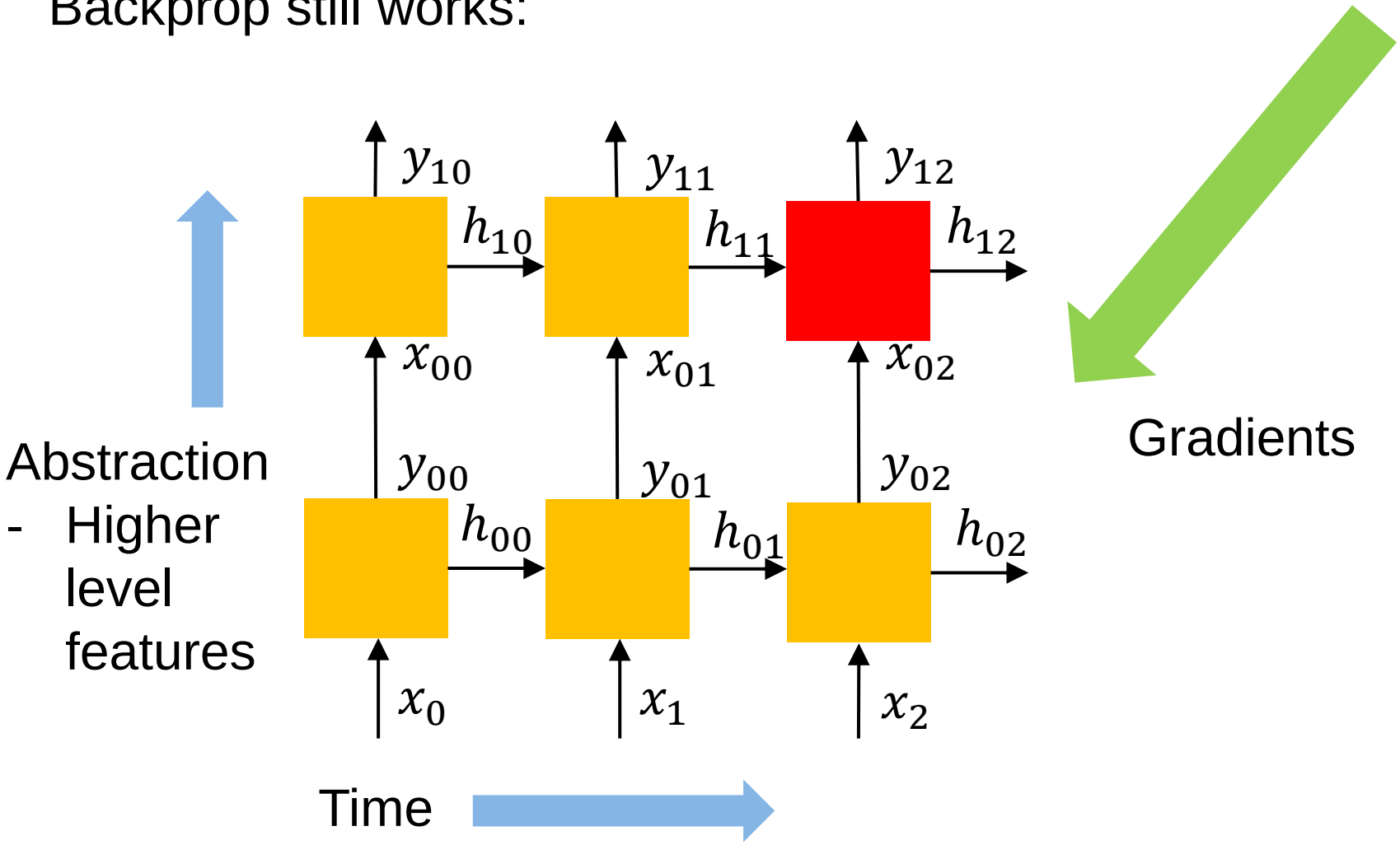
RNN structure

Backprop still works:



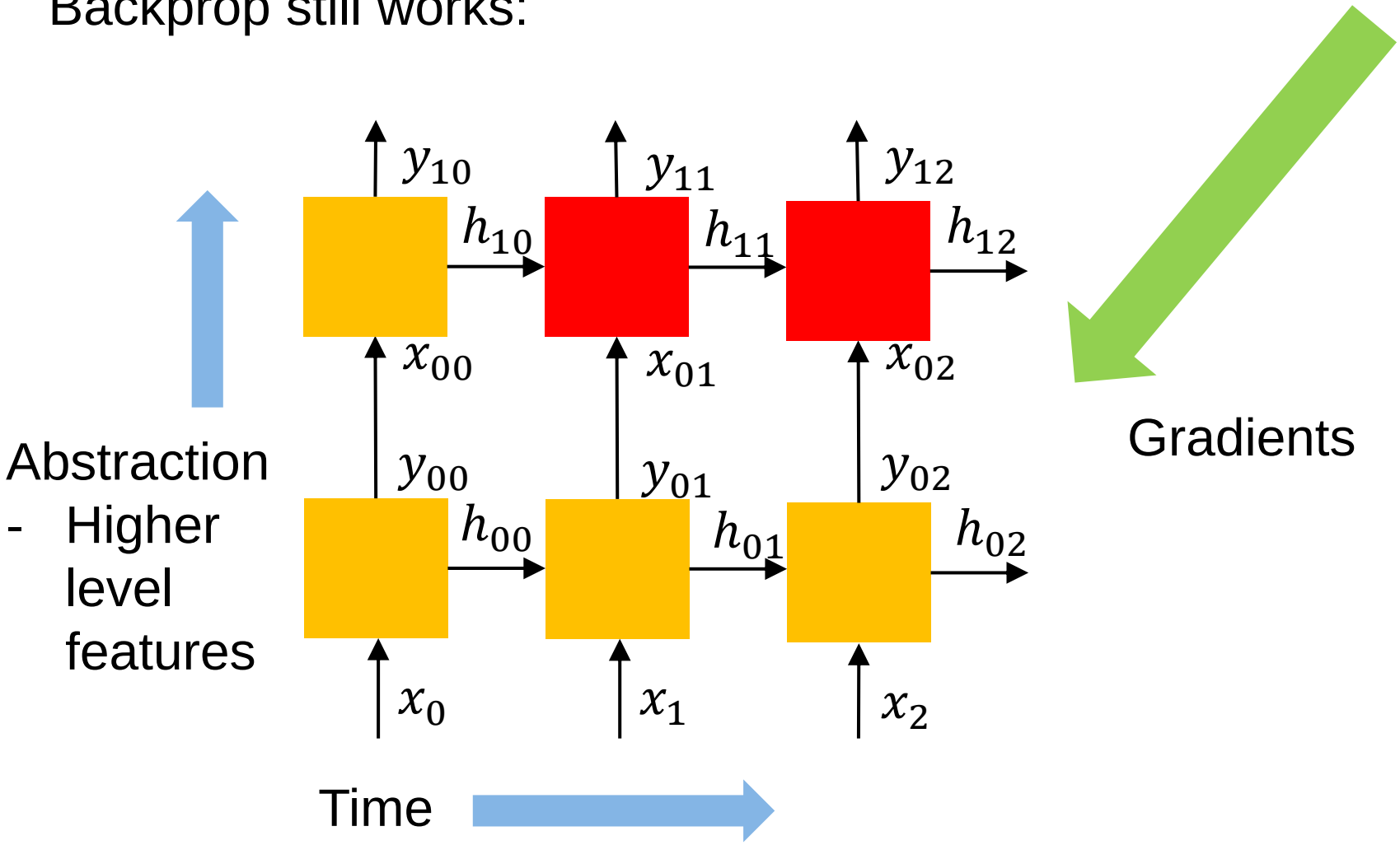
RNN structure

Backprop still works:



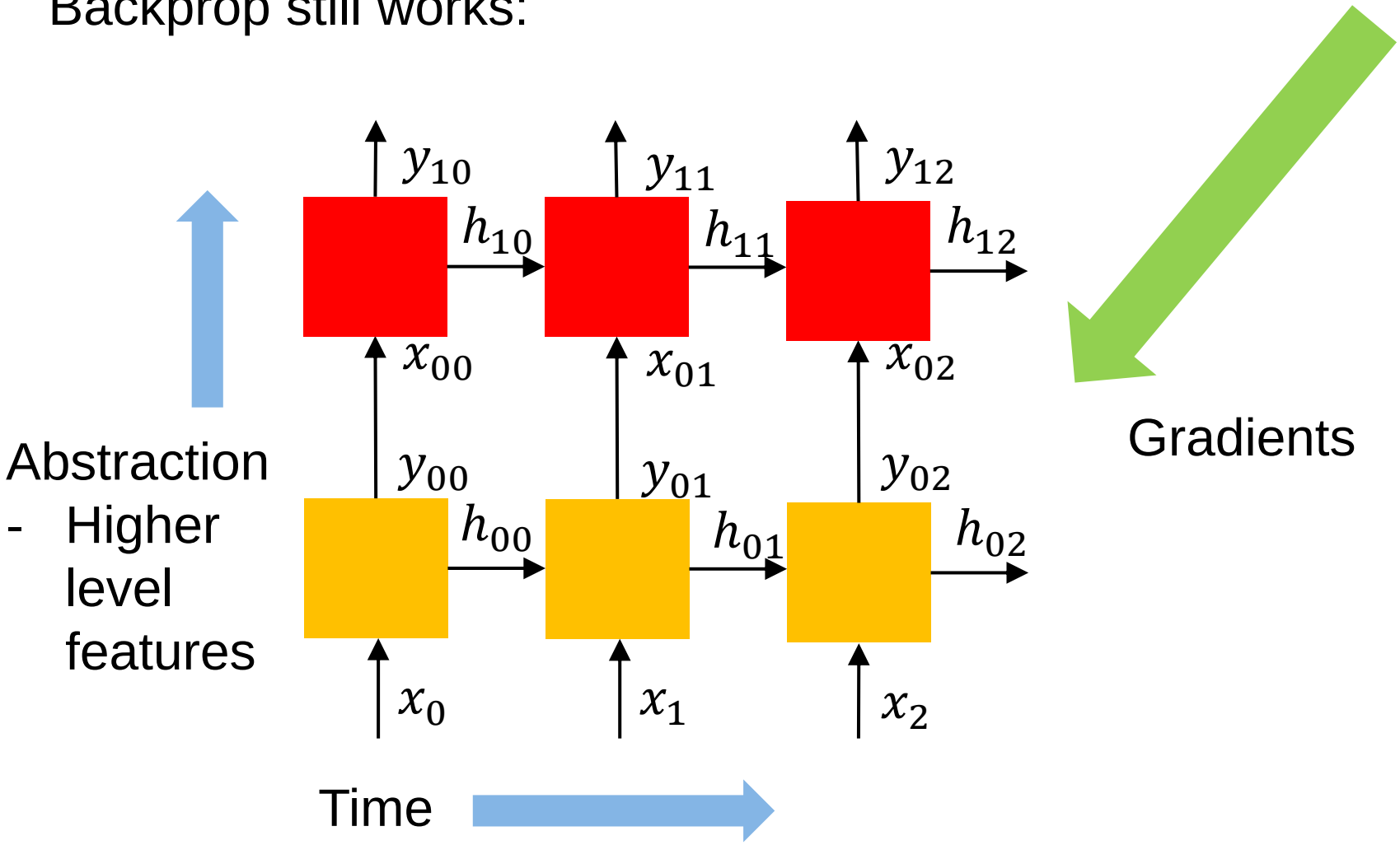
RNN structure

Backprop still works:



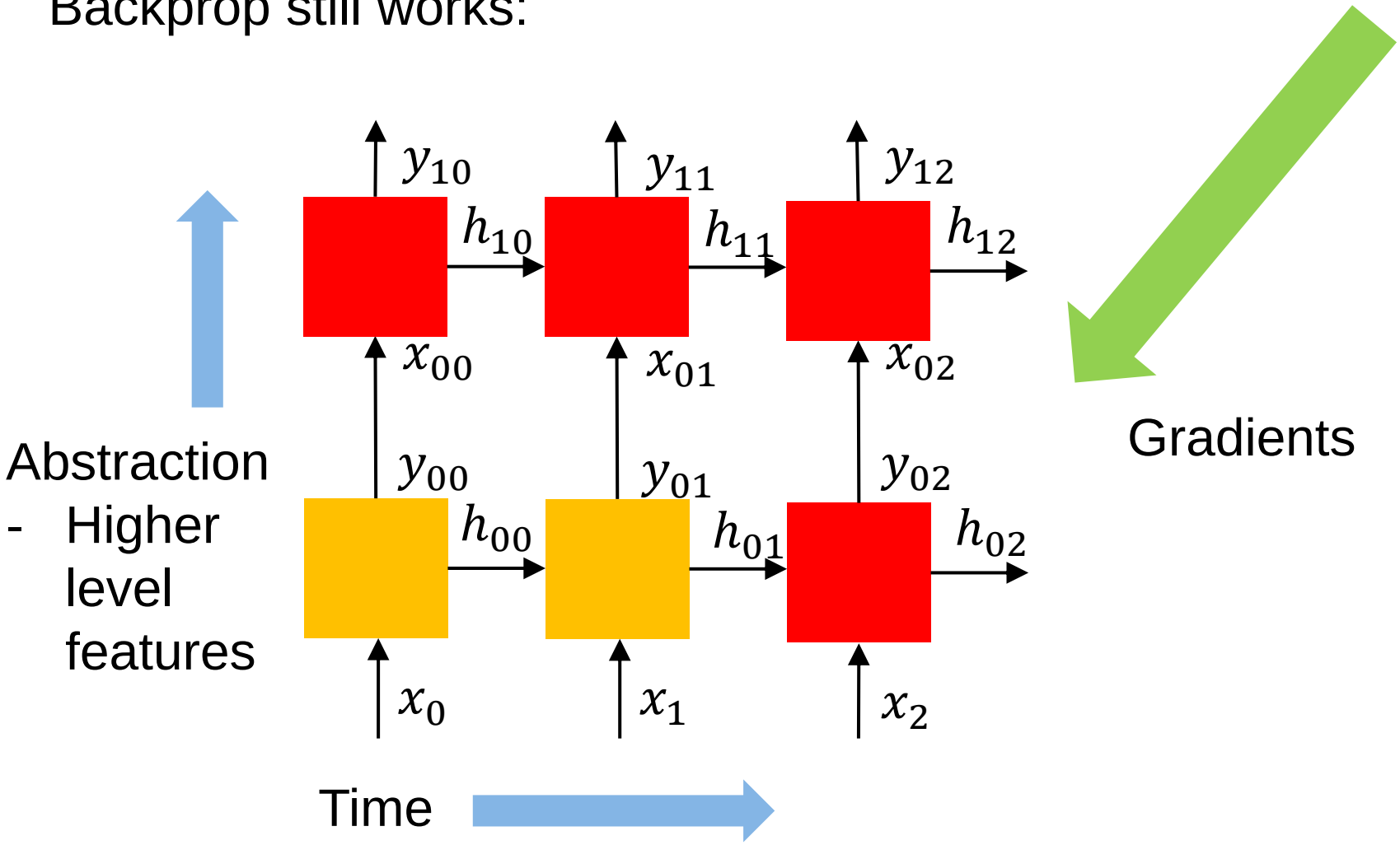
RNN structure

Backprop still works:



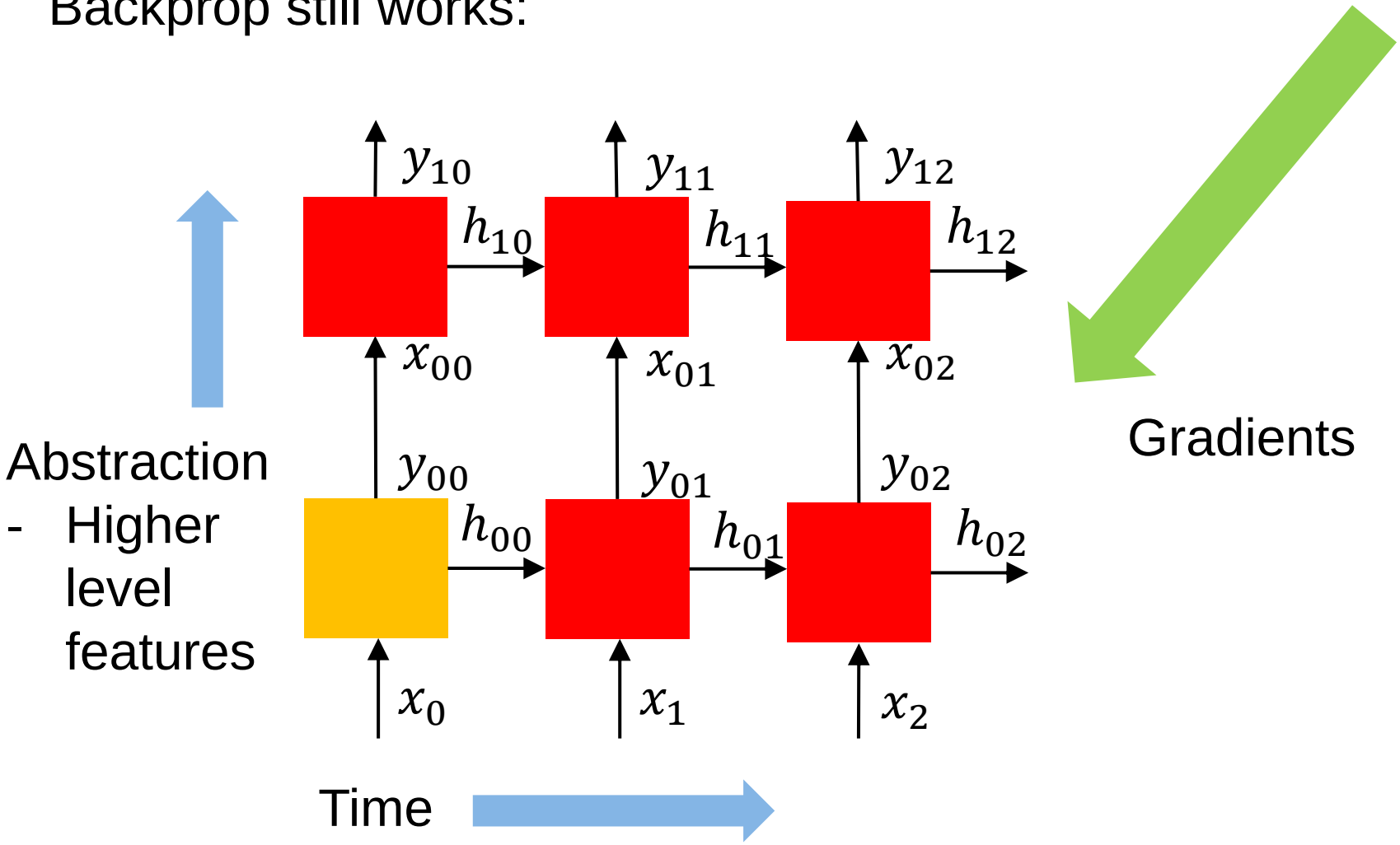
RNN structure

Backprop still works:



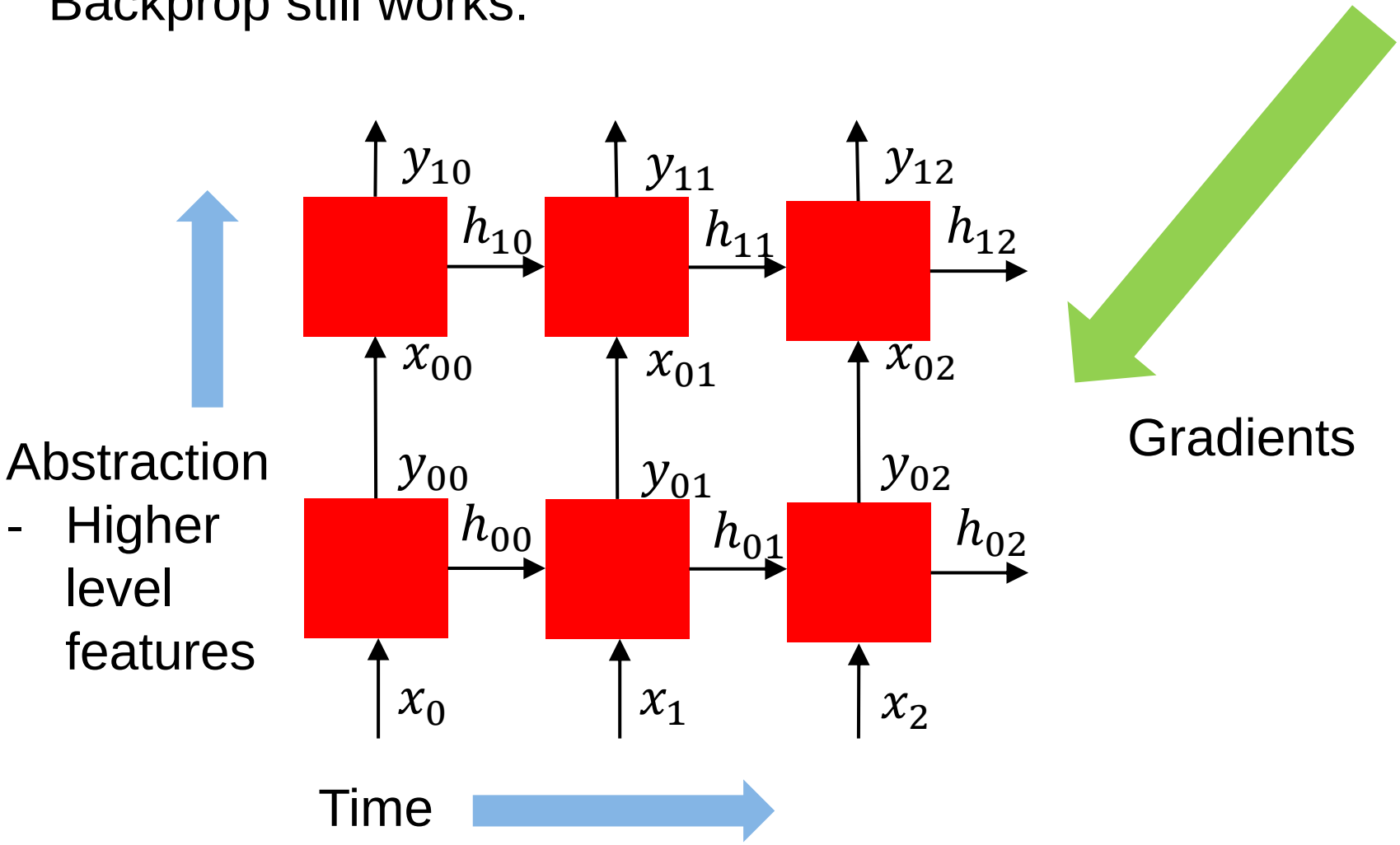
RNN structure

Backprop still works:



RNN structure

Backprop still works:



Recurrent Neural Network

AMRITA
VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

We can process a sequence of vectors **x** by applying a recurrence formula at every time step:

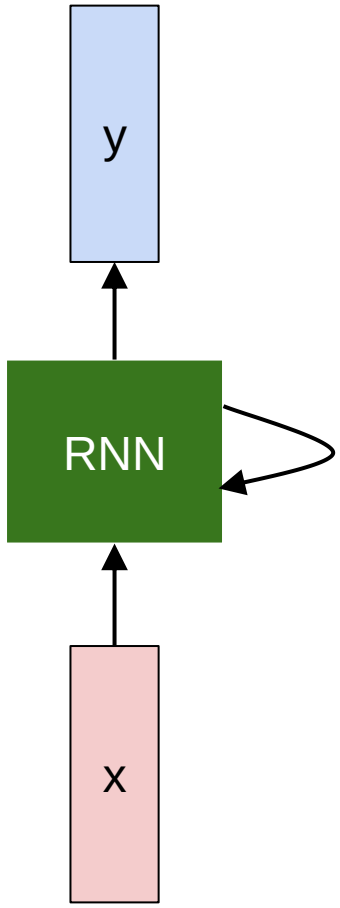
$$h_t = f_W(h_{t-1}, x_t)$$

new state

some function with parameters W

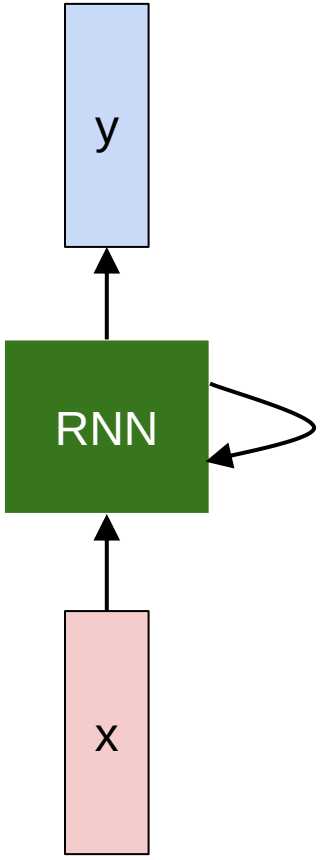
old state

input vector at some time step



Recurrent Neural Network

The state consists of a single *"hidden"* vector **h**:

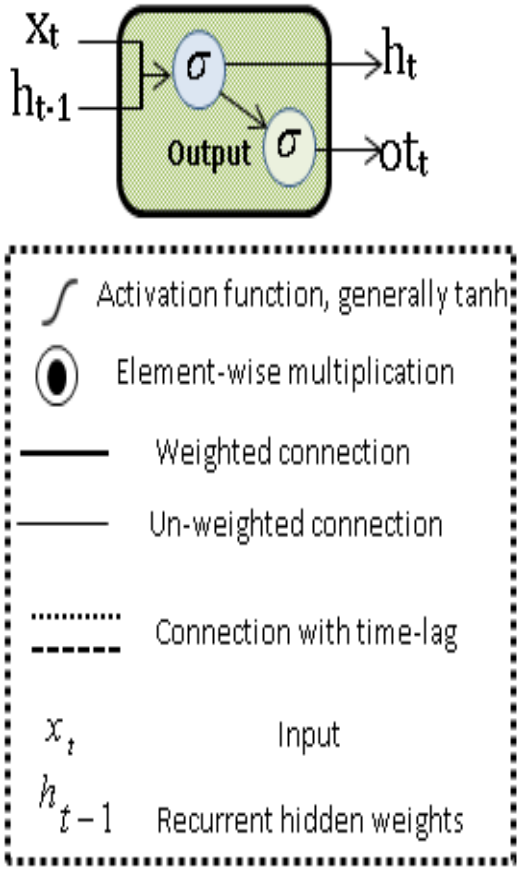


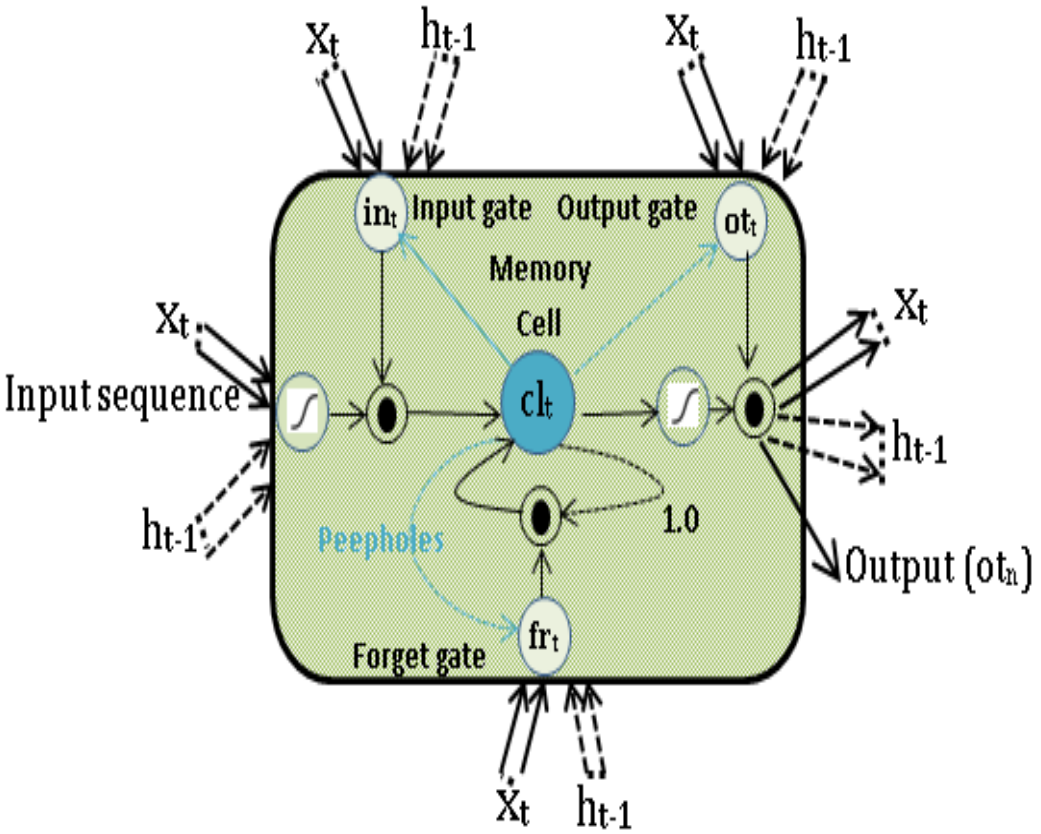
$$h_t = f_W(h_{t-1}, x_t)$$

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

$$y_t = W_{hy}h_t$$

Long short-term memory





Long short-term memory

AMRITA
VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

$$x_t, h_{t-1}, cl_{t-1} \rightarrow h_t, cl_t$$

$$in_t = \sigma(w_{xin} x_t + w_{hin} h_{t-1} + w_{clin} cl_{t-1} + b_{in})$$

$$fr_t = \sigma(w_{xfr} x_t + w_{hifr} h_{t-1} + w_{clfr} cl_{t-1} + b_{fr})$$

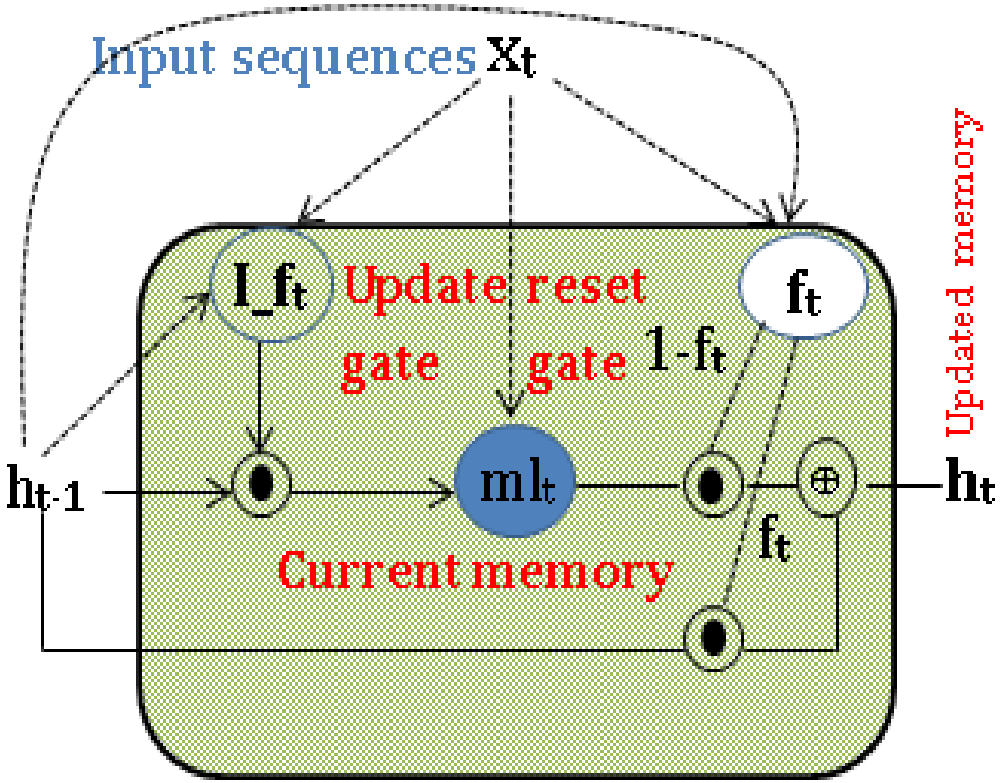
$$cl_t = fr_t \odot cl_{t-1} + in_t \odot \tanh(w_{xcl} x_t + w_{hcl} hi_{t-1} + b_{cl})$$

$$ot_t = \sigma(w_{xot} x_t + w_{hot} hi_{t-1} + w_{clot} cl_t + b_{ot})$$

$$h_t = ot_t \odot \tanh(cl_t)$$

Gated Recurrent Unit

Gated recurrent unit (GRU) is an alternative to LSTM networks.
Formulae shows, unlike LSTM memory cell with a list of gates (input, output and forget), GRU only consist of gates (update and forget) that are collectively involve in balancing the interior flow of information of the unit.



Gated Recurrent Unit

AMRITA
VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

$$x_t, h_{t-1} \rightarrow h_t$$

$$in\_fr_t = \sigma(w_{xin\_fr} x_t + w_{hiin\_fr} h_{t-1} + b_{in\_fr}) \quad \text{(Update gate)}$$

$$fr_t = \sigma(w_{xfr} x_t + w_{hifr} h_{t-1} + b_{fr}) \quad \text{(Forget or reset gate)}$$

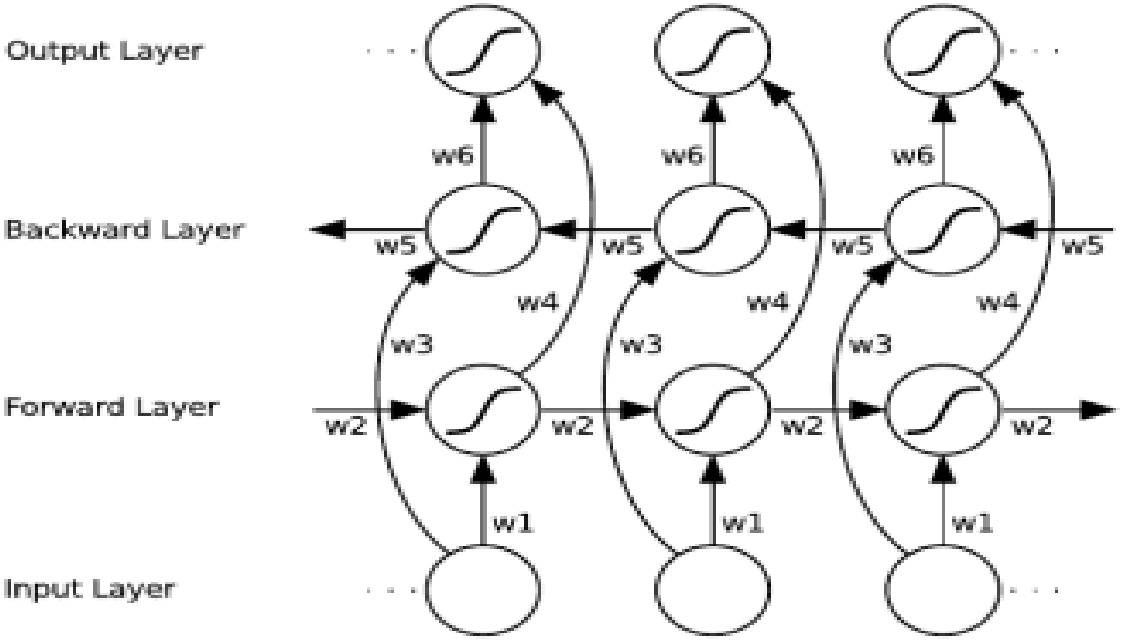
$$cl_t = \tanh(w_{xcl} x_t + w_{hcl}(fr \odot hi_{t-1}) + b_{cl}) \quad \text{(Current memory)}$$

$$h_t = f \odot h_{t-1} + (1-f) \odot cl \quad \text{(Updated memory)}$$

Extensions to LSTM architecture: Bidirectional RNN, LSTM, GRU

AMRITA VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

- Only the past information is taken into account in the training of a unidirectional RNN/LSTM
- Bidirectional architecture enables the use of future information
- Implementation with separate Forward-pass and Backward-pass specific layer weights
- Final output computed as the sum of forward and backward layer outputs



Summary

AMRITA
VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

- RNNs allow a lot of flexibility in architecture design
- RNNs are simple but don’t work very well
- Common to use LSTM or GRU: their additive interactions improve gradient flow
- Backward flow of gradients in RNN can explode or vanish. Exploding is controlled with gradient clipping. Vanishing is controlled with additive interactions (LSTM)
- Better/simpler architectures are a hot topic of current research
- Better understanding (both theoretical and empirical) is needed.

AMRITA
VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

Thank you

Questions ?

vinayakumarr77@gmail.com
https://vinayakumarr.github.io/
https://sites.google.com/site/vinayakumarr77/

Software Installation

AMRITA
VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

- sudo apt-get install libatlas-base-dev gfortran python-dev
- sudo apt-get install python-pip
- sudo pip install --upgrade pip
- sudo pip install numpy
- sudo pip install scipy
- sudo pip install matplotlib
- Sudo pip install seaborn
- sudo pip install scikit-learn
- sudo pip install tensorflow
- sudo pip install theano
- sudo pip install keras
- sudo pip install pandas
- sudo pip install h5py
- sudo pip install jupyter
- sudo pip install ipython