COMP7/8118 M50

Data Mining

Recurrent Neural Network: Basic Model

Xiaofei Zhang

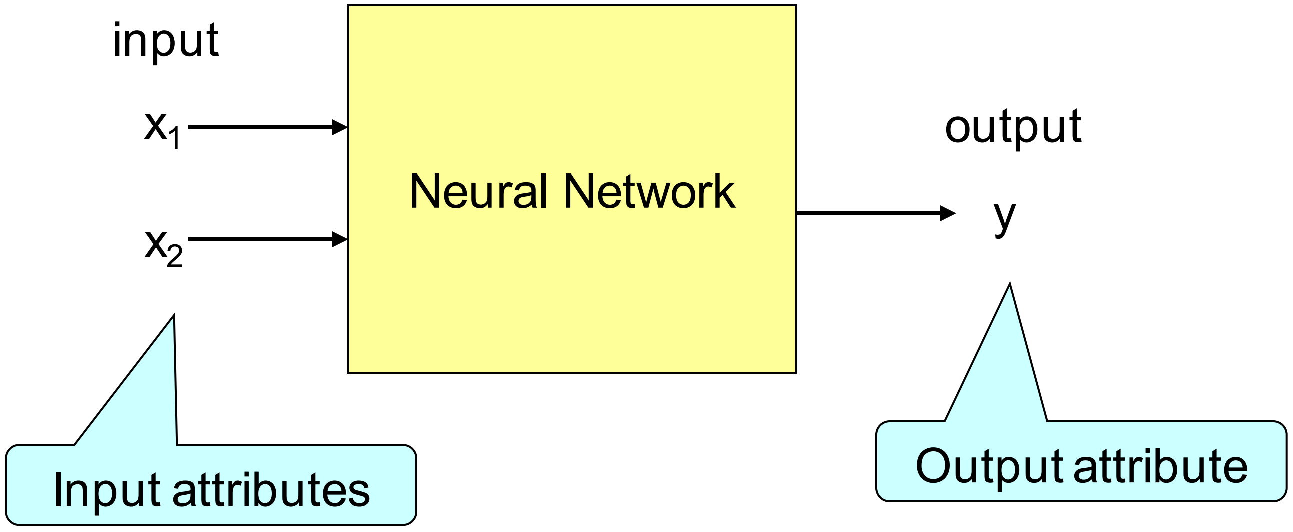
Slides compiled from Jiawei Han and Raymond C.W. Wong's work

THE UNIVERSITY OF MEMPHIS

Neural Network

- Neural Network

x_1	x_2	d
0	0	0
0	1	1
1	0	1
1	1	1



Neural Network

- Here, we know that training the model with one record is "independent" of training the model with another record

- This means that we assume that records in the table are "independent"

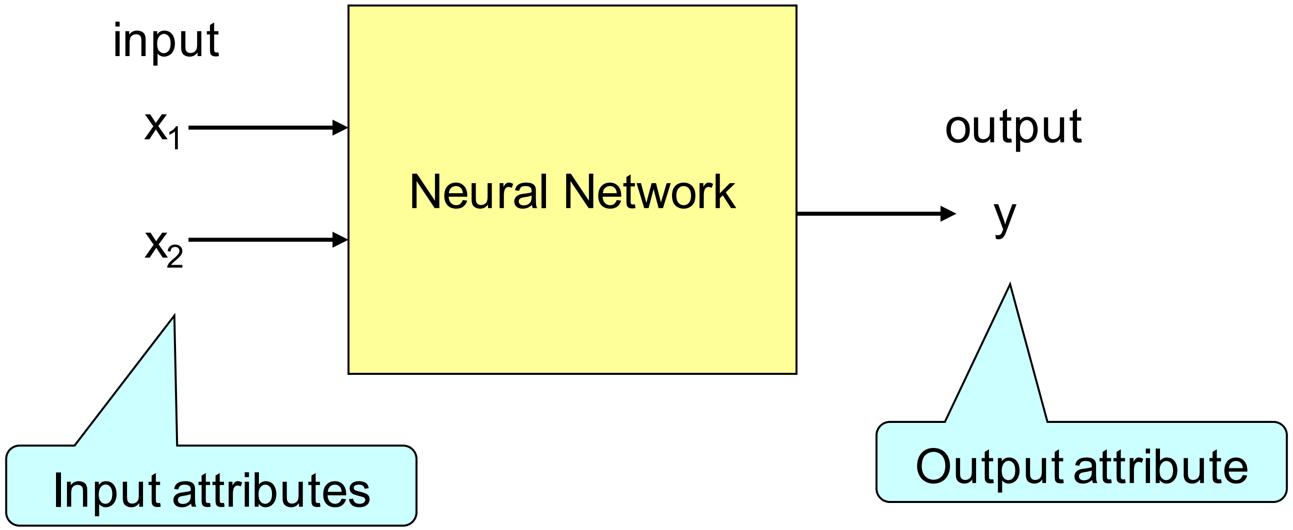
Neural Network

- In some cases, the current record is “related” to the “previous” records in the table.
- Thus, records in the table are “dependent”.
- We also want to capture this “dependency” in the model

We could use a new model called “recurrent neural network” for this purpose.

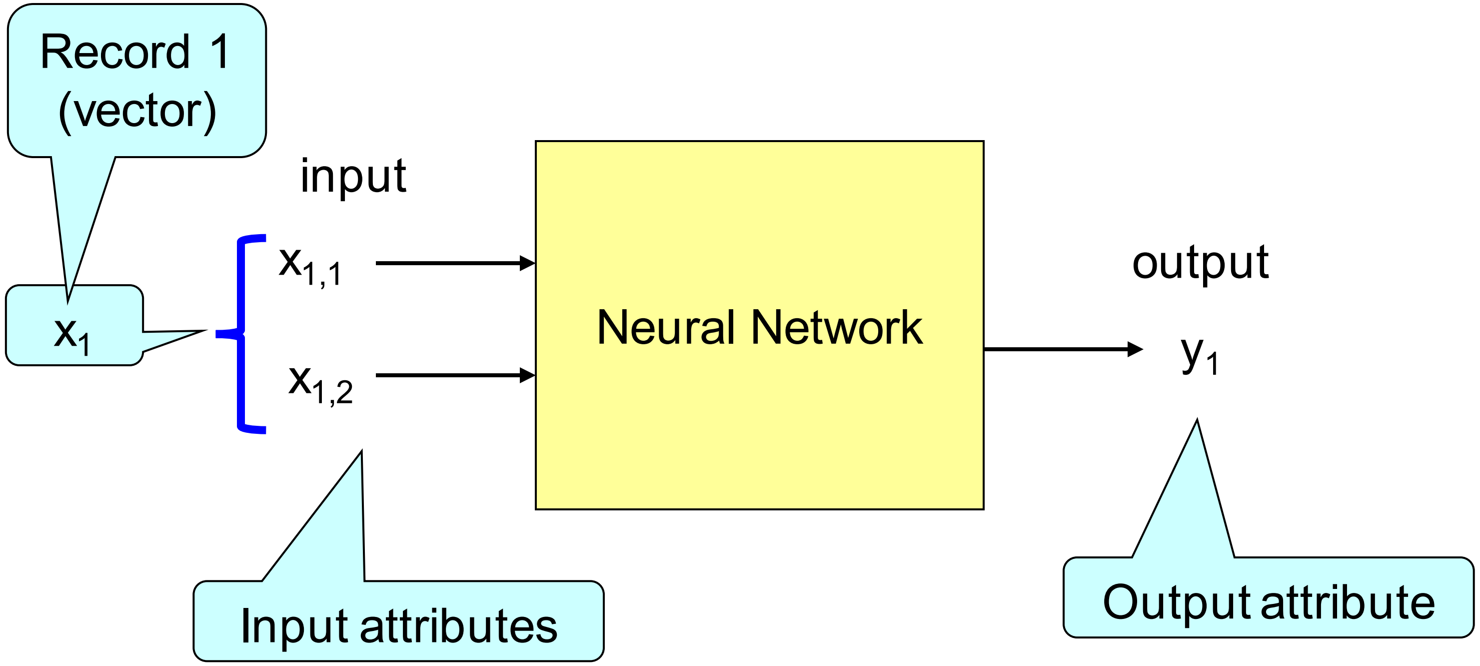
Neural Network

- Neural Network



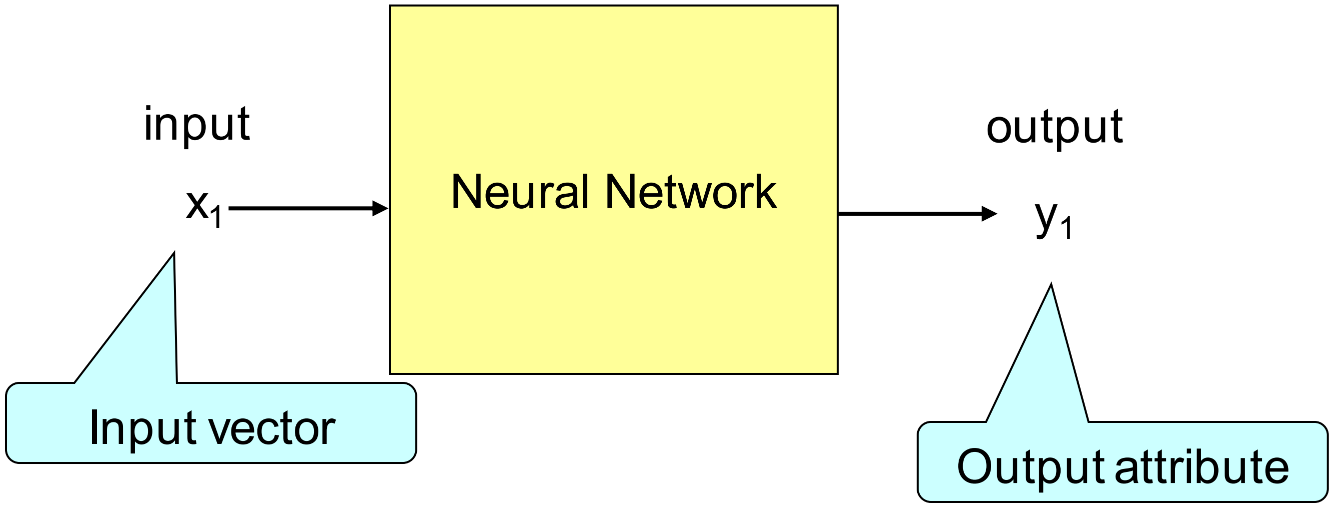
Neural Network

- Neural Network



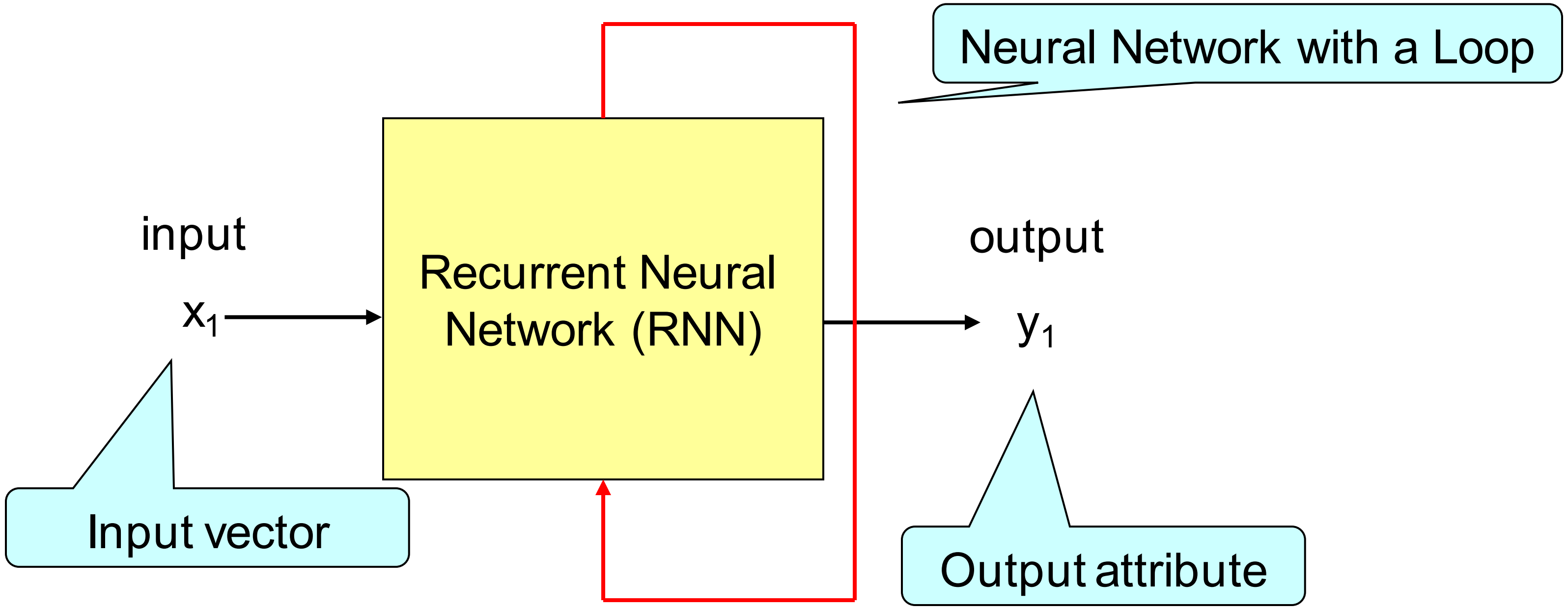
Neural Network

- Neural Network

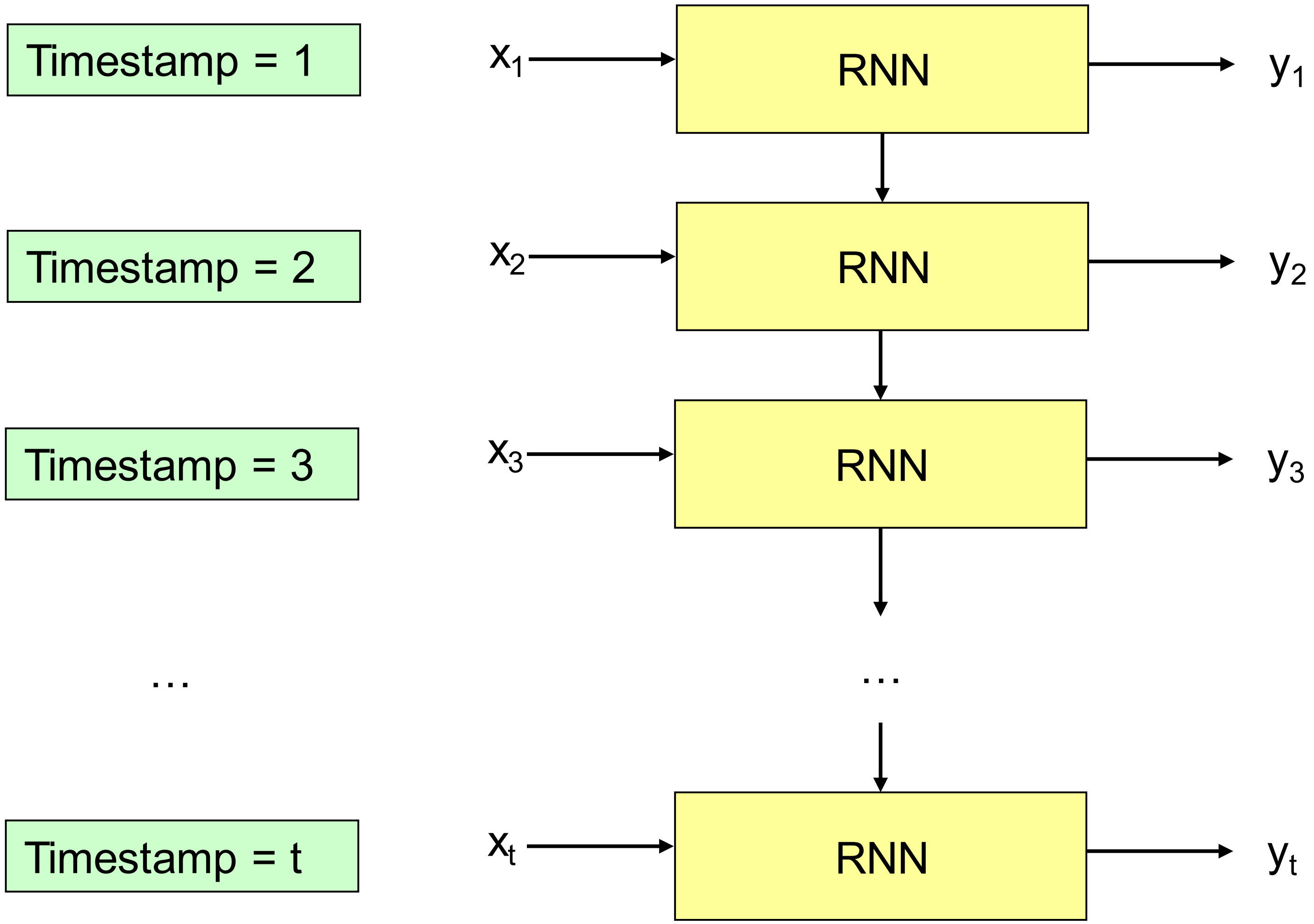


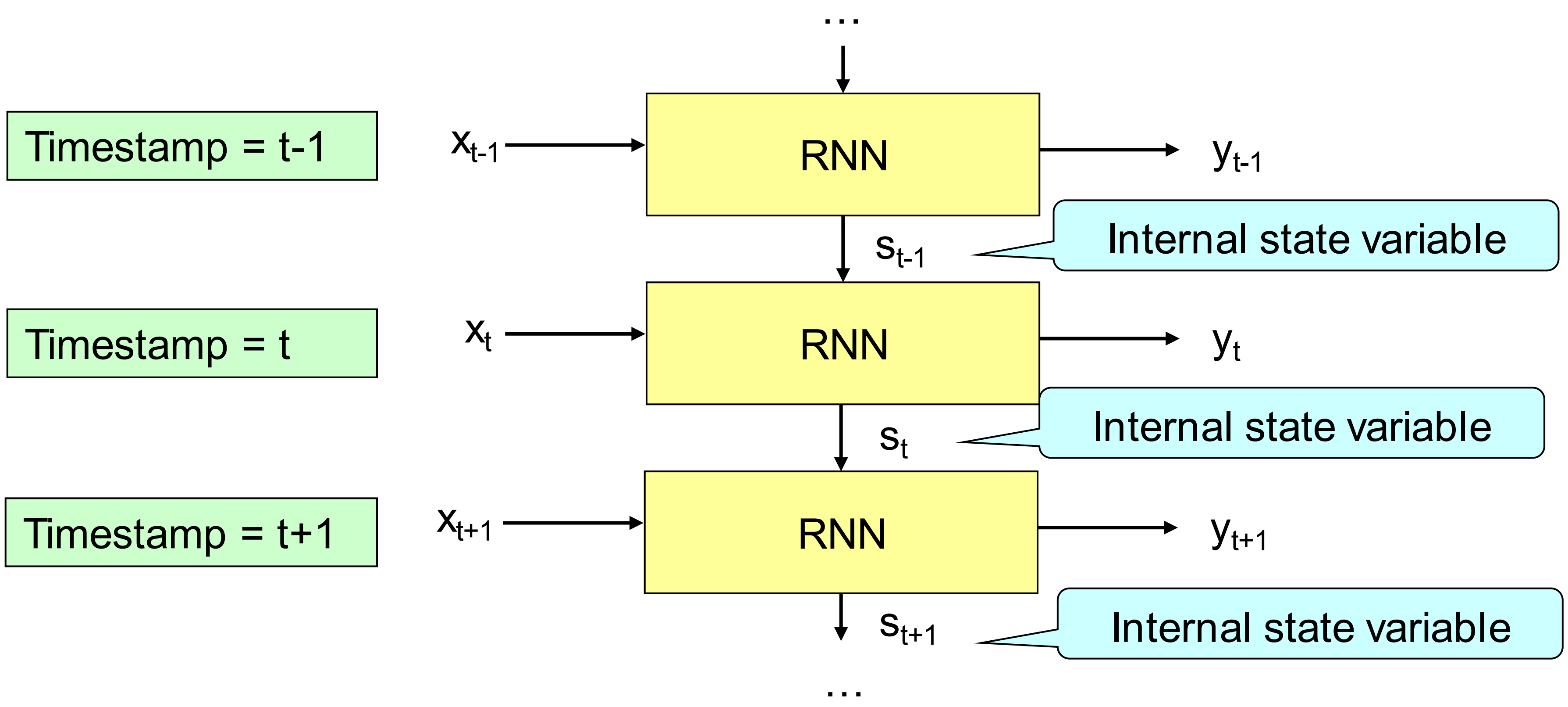
Recurrent Neural Network

- Recurrent Neural Network (RNN)



Unfolded representation of RNN




...
Timestamp = t-1
x_{t-1}
RNN
y_{t-1}
s_{t-1}
Internal state variable
Timestamp = t
x_t
RNN
y_t
s_t
Internal state variable
Timestamp = t+1
x_{t+1}
RNN
y_{t+1}
s_{t+1}
Internal state variable
...

Recurrent Neural Network

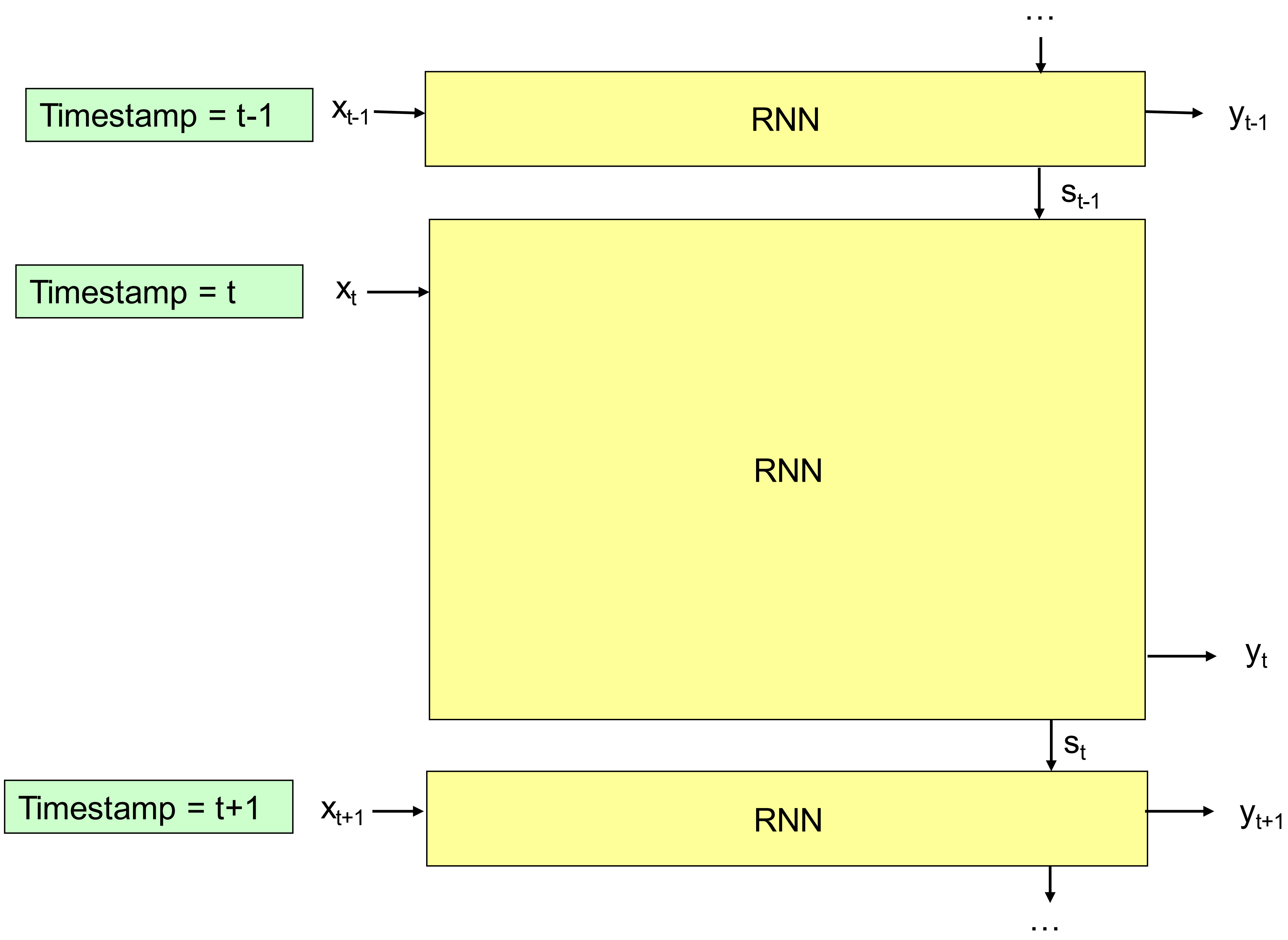
- Limitation
  - It may “memorize” a lot of past events/values
  - Due to its complex structure, it is more time-consuming for training.

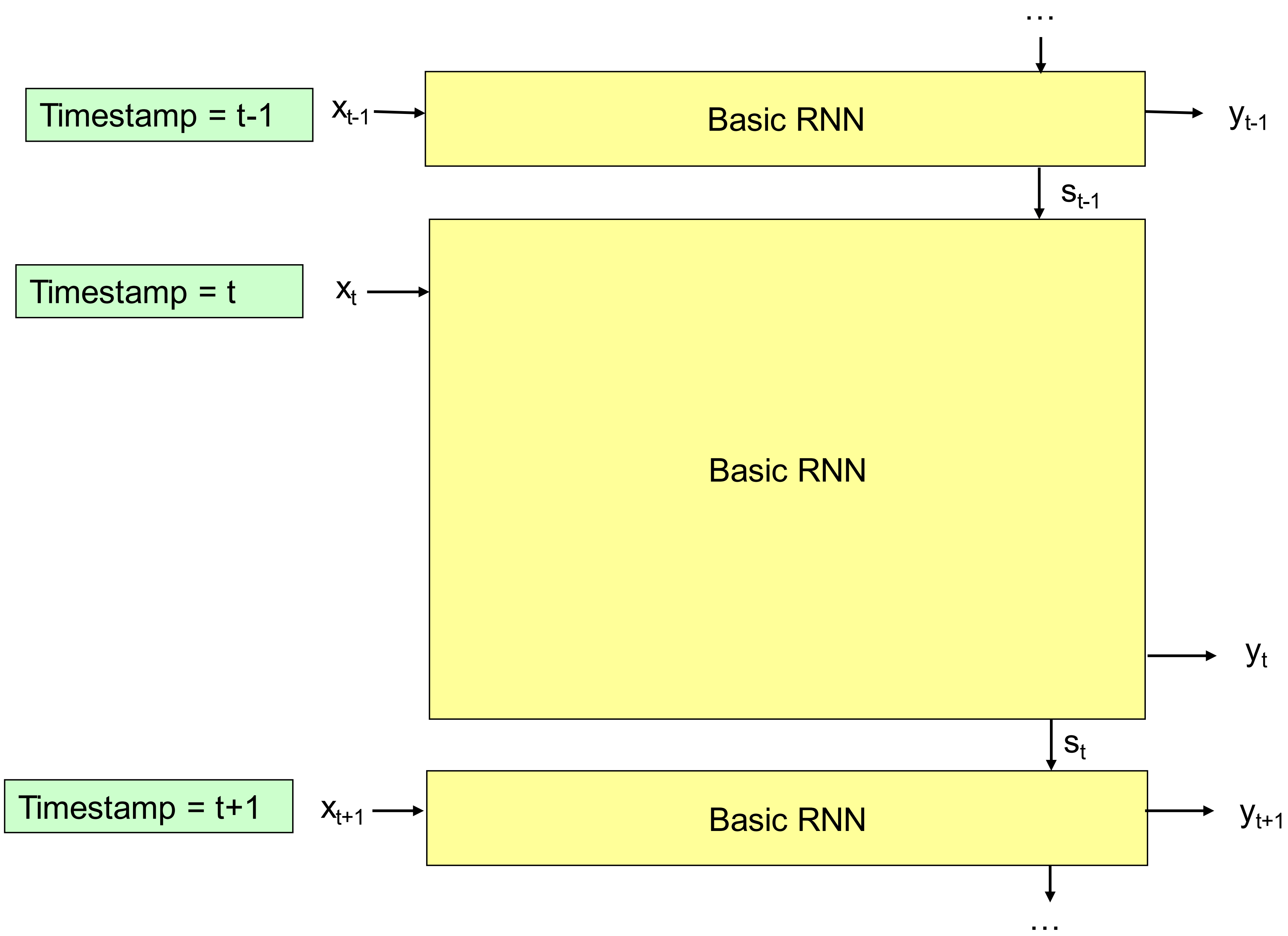
RNN

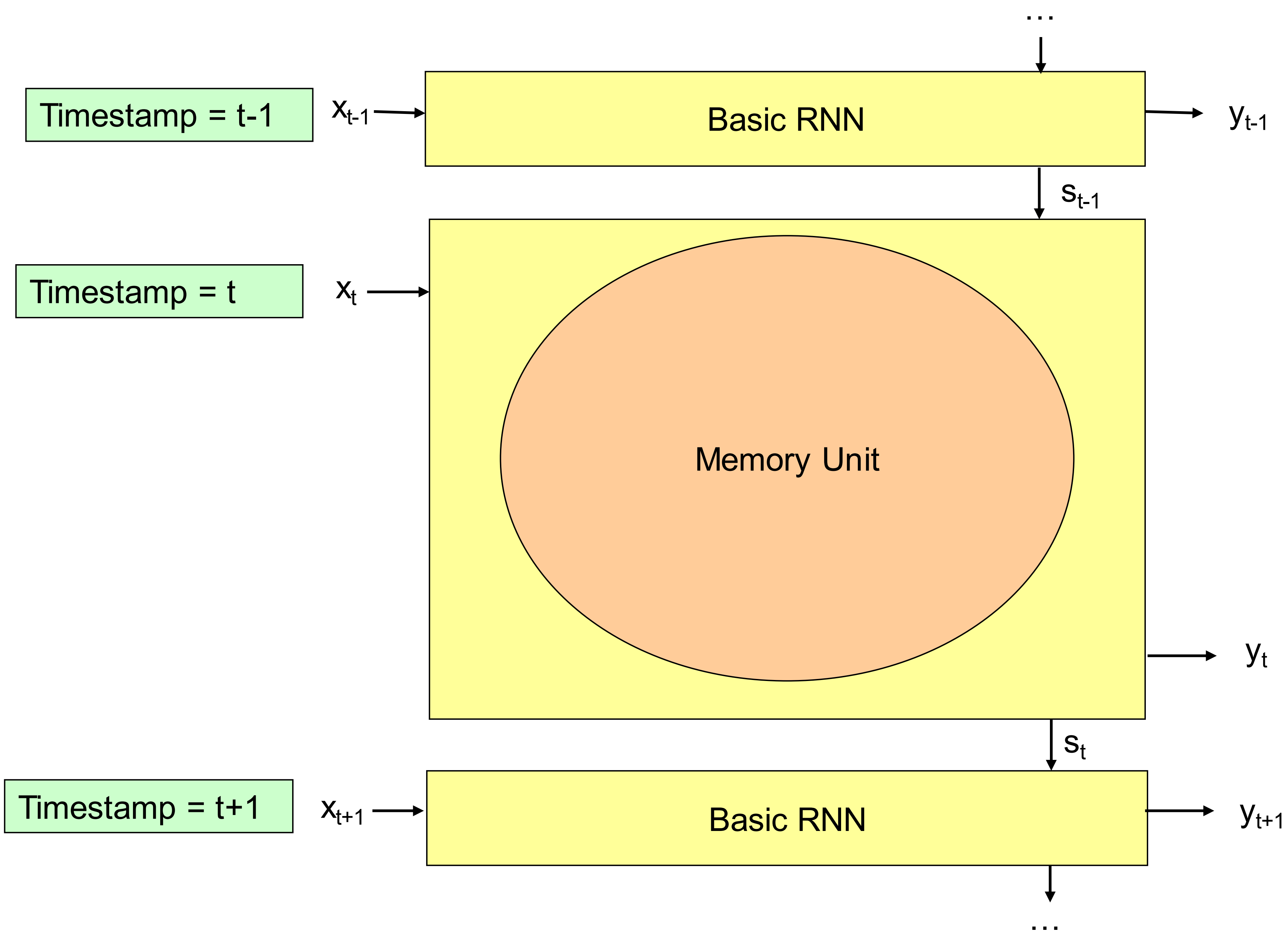
1. Basic RNN
2. Traditional LSTM
3. GRU

Basic RNN

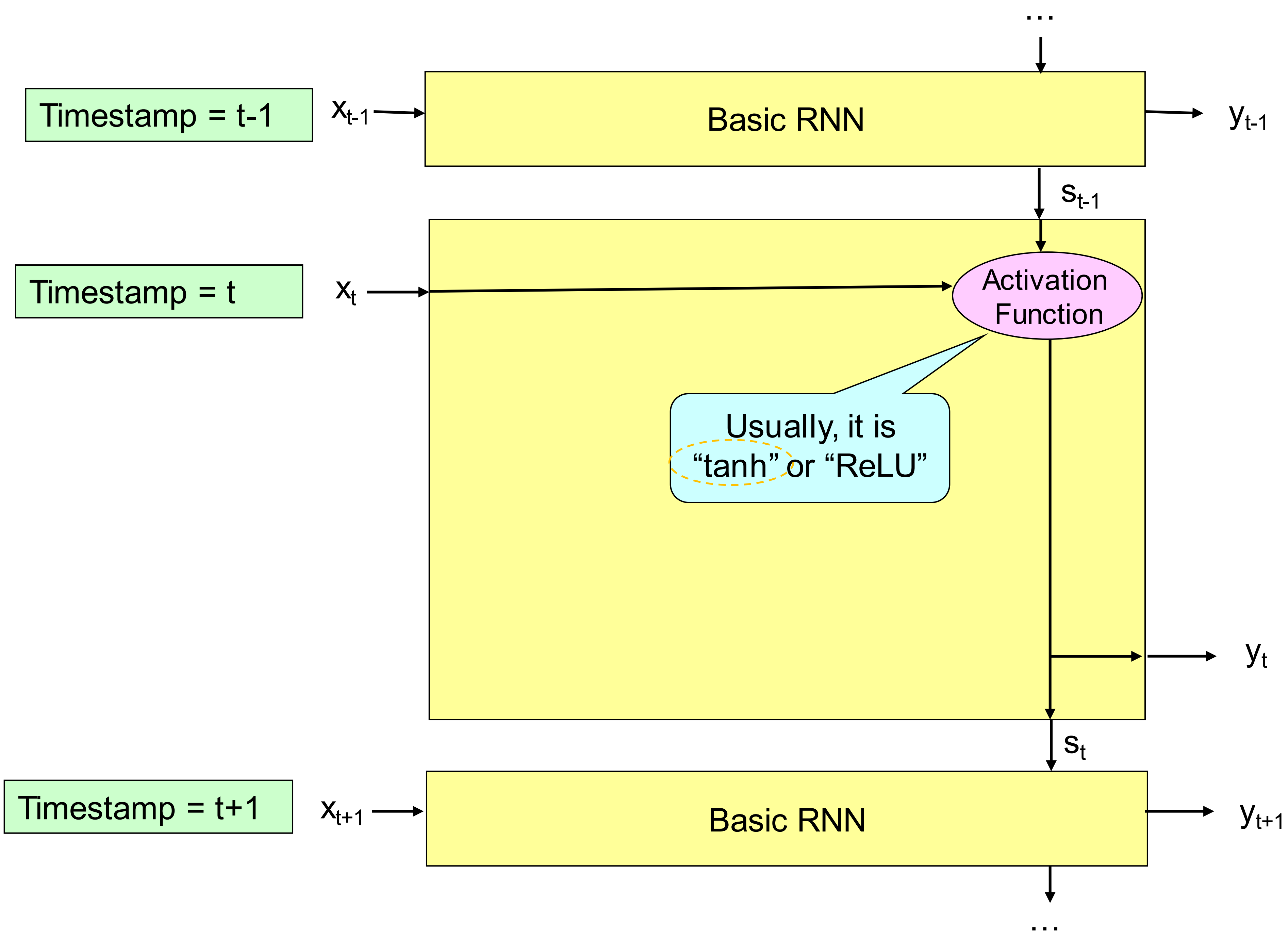
- The basic RNN is very simple.
- It contains only one single activation function (e.g., “tanh” and “ReLU”).


Timestamp = t-1
Timestamp = t
Timestamp = t+1
x_{t-1}
x_t
x_{t+1}
RNN
RNN
RNN
y_{t-1}
y_t
y_{t+1}
s_{t-1}
s_t
…
…

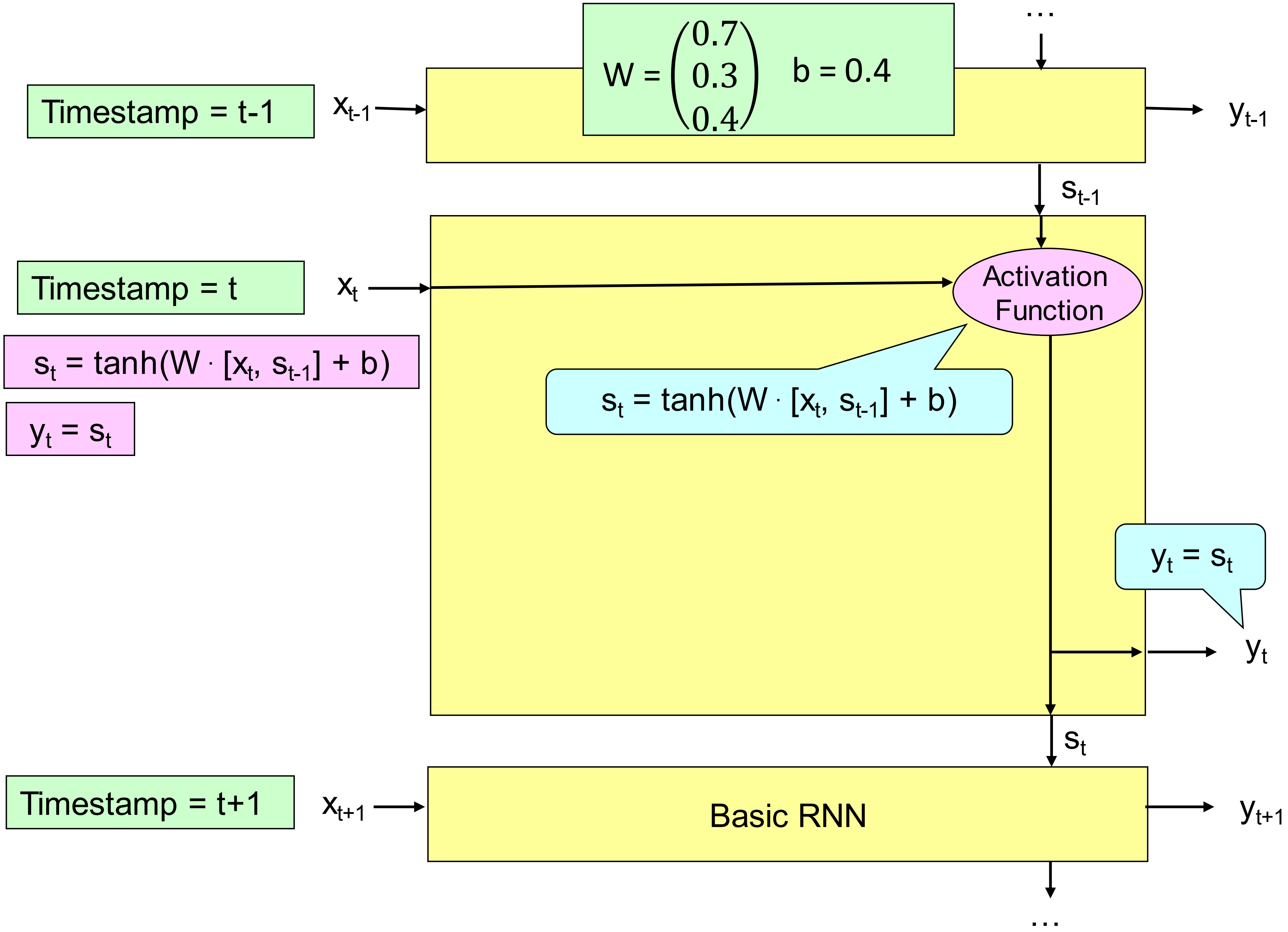

Timestamp = t-1
Timestamp = t
Timestamp = t+1
x_{t-1}
x_t
x_{t+1}
Basic RNN
Basic RNN
Basic RNN
y_{t-1}
y_t
y_{t+1}
s_{t-1}
s_t
…
…



...
Timestamp = t-1
x_{t-1}
Basic RNN
y_{t-1}
s_{t-1}
Timestamp = t
x_t
Memory Unit
y_t
s_t
Timestamp = t+1
x_{t+1}
Basic RNN
y_{t+1}
...



...
Timestamp = t-1
x_{t-1}
Basic RNN
y_{t-1}
s_{t-1}
Timestamp = t
x_t
Activation Function
Usually, it is "tanh" or "ReLU"
y_t
s_t
Timestamp = t+1
x_{t+1}
Basic RNN
y_{t+1}
...

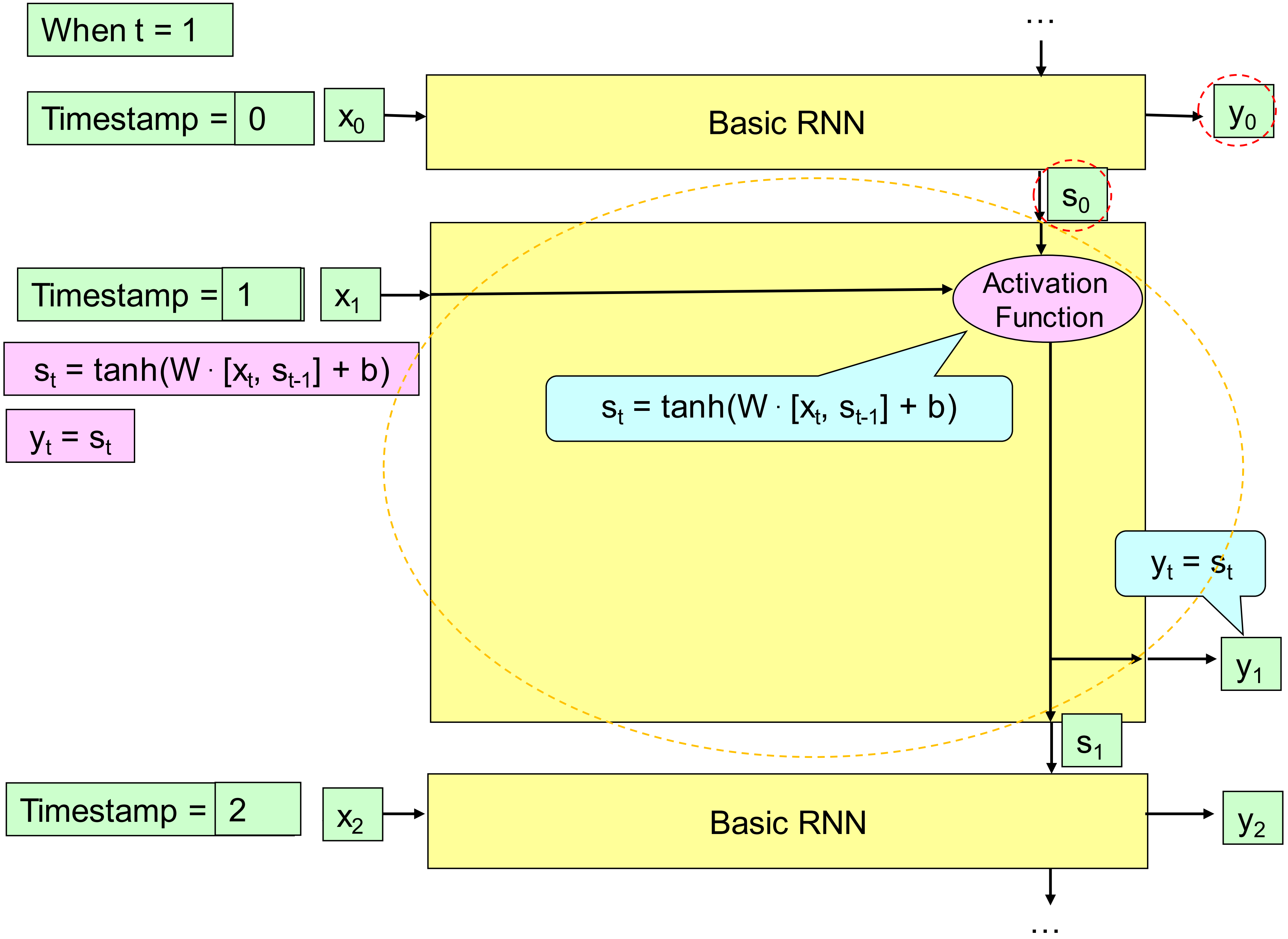

Timestamp = t-1
x_{t-1}
$W = \begin{pmatrix} 0.7 \\ 0.3 \\ 0.4 \end{pmatrix}$ $b = 0.4$
...
y_{t-1}
s_{t-1}
Timestamp = t
x_t
Activation Function
$s_t = \tanh(W \cdot [x_t, s_{t-1}] + b)$
$y_t = s_t$
$s_t = \tanh(W \cdot [x_t, s_{t-1}] + b)$
$y_t = s_t$
y_t
s_t
Timestamp = t+1
x_{t+1}
Basic RNN
y_{t+1}
...

Basic RNN

- In the following, we want to compute (weight) values in the basic RNN.
- Similar to the neural network, the basic RNN model has two steps.
  - **Step 1** (Input Forward Propagation)
  - **Step 2** (Error Backward Propagation)
- In the following, we focus on “Input Forward Propagation”.
- In the basic RNN, “Error Backward Propagation” could be solved by an existing optimization tool (like “Neural Network”).

Time	$x_{t, 1}$	$x_{t, 2}$	y
t=1	0.1	0.4	0.3
t=2	0.7	0.9	0.5

- Consider this example with two timestamps.
  - When t = 1
  - When t = 2
- We use the basic RNN to do the training.


When t = 1
Timestamp = 0
x0
Basic RNN
y0
s0
Timestamp = 1
x1
Activation Function
s_t = tanh(W · [x_t, s_t-1] + b)
y_t = s_t
s_t = tanh(W · [x_t, s_t-1] + b)
y_t = s_t
y1
s1
Timestamp = 2
x2
Basic RNN
y2
…
…

$s_t = \tanh(W \cdot [x_t, s_{t-1}] + b)$

$y_t = s_t$

Time	$x_{t,1}$	$x_{t,2}$	y
t=1	0.1	0.4	0.3
t=2	0.7	0.9	0.5

y_0	s_0
0	0

Step 1 (Input Forward Propagation)

$s_1 = \tanh(W \cdot [x_1, s_0] + b)$

$= \tanh(\begin{pmatrix} 0.7 \\ 0.3 \\ 0.4 \end{pmatrix} \begin{pmatrix} 0.1 \\ 0.4 \\ 0 \end{pmatrix} + 0.4)$

$= \tanh(0.7 \cdot 0.1 + 0.3 \cdot 0.4 + 0.4 \cdot 0 + 0.4)$

$= \tanh(0.59)$

$= 0.5299$

$W = \begin{pmatrix} 0.7 \\ 0.3 \\ 0.4 \end{pmatrix}$ $b = 0.4$

$s_1 = 0.5299$

$y_1 = 0.5299$

$y_1 = s_1$

$= 0.5299$

Error $= y_1 - y$

$= 0.5299 - 0.3$

$= 0.2299$

$s_t = \tanh(W \cdot [x_t, s_{t-1}] + b)$

$y_t = s_t$

Step 1 (Input Forward Propagation)

Time	$x_{t,1}$	$x_{t,2}$	y
t=1	0.1	0.4	0.3
t=2	0.7	0.9	0.5

y_1	s_1
0.5299	0.5299

$W = \begin{pmatrix} 0.7 \\ 0.3 \\ 0.4 \end{pmatrix}$ $b = 0.4$

$s_1 = 0.5299$

$y_1 = 0.5299$

$s_t = \tanh(W \cdot [x_t, s_{t-1}] + b)$

$y_t = s_t$

Time	$x_{t,1}$	$x_{t,2}$	y
t=1	0.1	0.4	0.3
t=2	0.7	0.9	0.5

y_1	s_1
0.5299	0.5299

$$W = \begin{pmatrix} 0.7 \\ 0.3 \\ 0.4 \end{pmatrix} \quad b = 0.4$$

Step 1 (Input Forward Propagation)

$s_2 = \tanh(W \cdot [x_2, s_1] + b)$

$$= \tanh(\begin{pmatrix} 0.7 \\ 0.3 \\ 0.4 \end{pmatrix} \begin{pmatrix} 0.7 \\ 0.9 \\ 0.5299 \end{pmatrix} + 0.4)$$

$= \tanh(0.7 \cdot 0.7 + 0.3 \cdot 0.9 + 0.4 \cdot 0.5299 + 0.4)$

$= \tanh(1.3720)$

$= 0.8791$

$y_2 = s_2$

$= 0.8791$

Error $= y_2 - y$

$= 0.8791 – 0.5$

$= 0.3791$

$s_2 = 0.8791$

$y_2 = 0.8791$