

# Recurrent Neural Networks

Machine Learning for Behavioral Data

April 17, 2023

# Today's Topic

Week	Lecture/Lab
8	Spring Break
9	Time Series Prediction
10	Unsupervised Learning
11	Unsupervised Learning
12	Ethical Machine Learning
13	Ethical Machine Learning
14	Project Presentations
15	Whit Monday

**Supervised learning on time series:**

- Probabilistic graphical models
- Neural networks: LSTM, GRU, etc.

# Getting ready for today's lecture...

- **If not done yet:** clone the repository containing the Jupyter notebook and data for today's lecture into your Noto workspace.
- SpeakUp room for today's lecture:

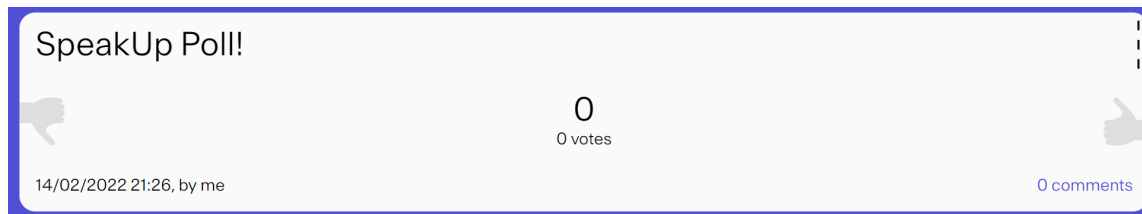
<https://go.epfl.ch/speakup-mlbd>



# Short quiz about the past...

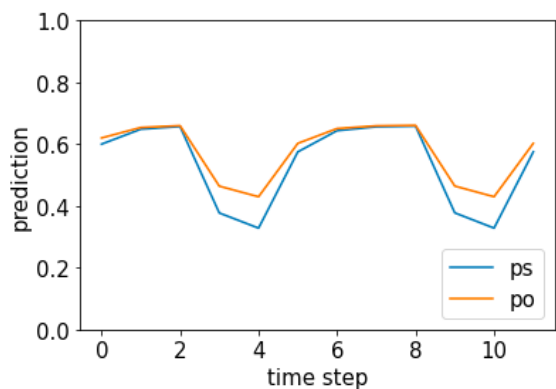
This KT model uses the # of opportunities the student had per skill and treats prior successes and failures the same.

- a) Additive Factors Model (AFM)
- b) Performance Factors Analysis (PFA)
- c) Bayesian Knowledge Tracing (BKT)

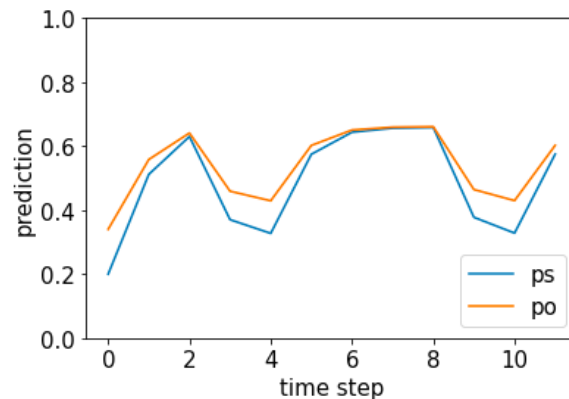


# Short quiz about the past...

Which BKT parameter has been changed between the left and right plot (exactly one)?



$p_0 = 0.6,$   
 $p_s = 0.1,$   
 $p_g = 0.2,$   
 $p_l = 0.3,$   
 $p_f = 0.3$



a)  $p_g$  (guess probability)

b)  $p_l$  (probability of learning)

c)  $p_0$  (initial probability)

d)  $p_f$  (forget probability)

SpeakUp Poll!

0

0 votes

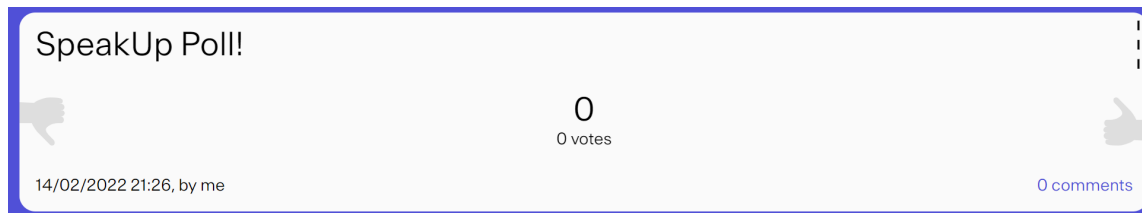
14/02/2022 21:26, by me

0 comments

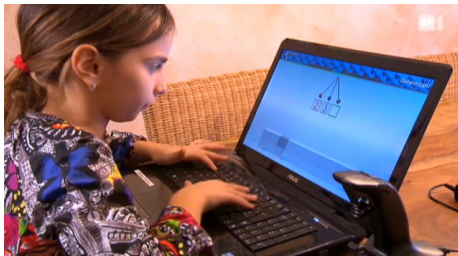
# Short quiz about the past...

Which of the following statements about Pearson's correlation is true?

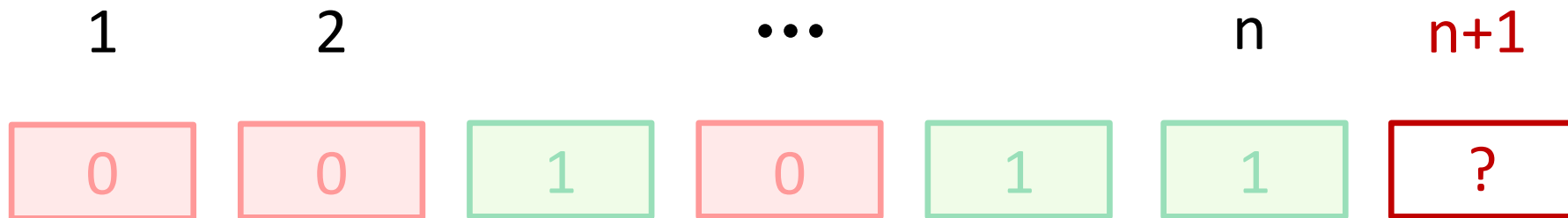
- a) If two variables  $X, Y$  have correlation  $= 0$ , then  $X, Y$  are dependent.
- b) If two variables  $X, Y$  have correlation  $= 0$ , then  $X, Y$  are independent.
- c) If  $X, Y$  are dependent variables, then their correlation  $= 0$ .
- d) If  $X, Y$  are independent variables, then their correlation  $= 0$ .



# Knowledge Tracing – Predicting Future Performance



## Subtraction 0-100



# Today – Recurrent Neural Networks

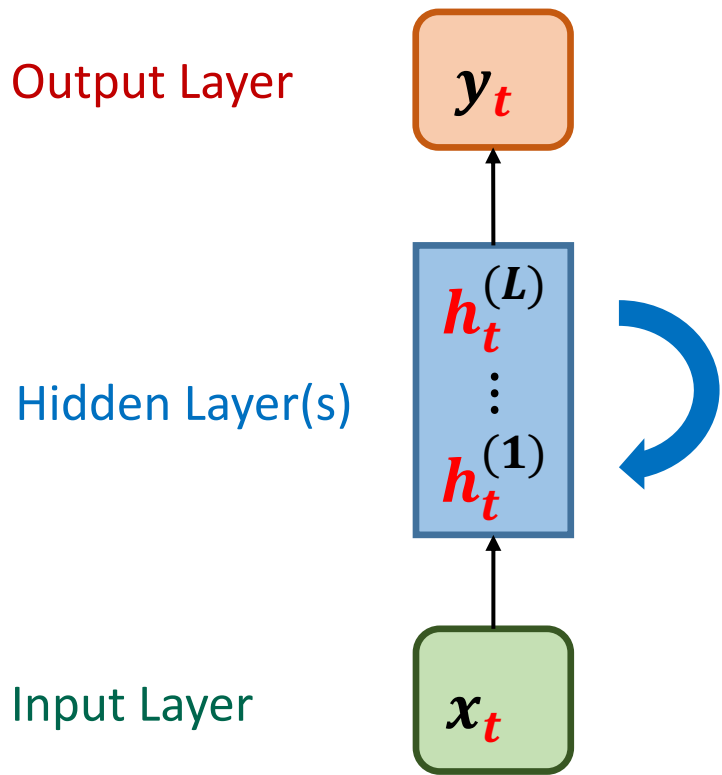
- **Deep Knowledge Tracing**
  - Parameters and hyperparameter tuning
  - Different architectures
  - Different tasks:
    - “Many-to-many” versus “Many-to-one”
    - Classification versus Regression
-



# Neural Networks

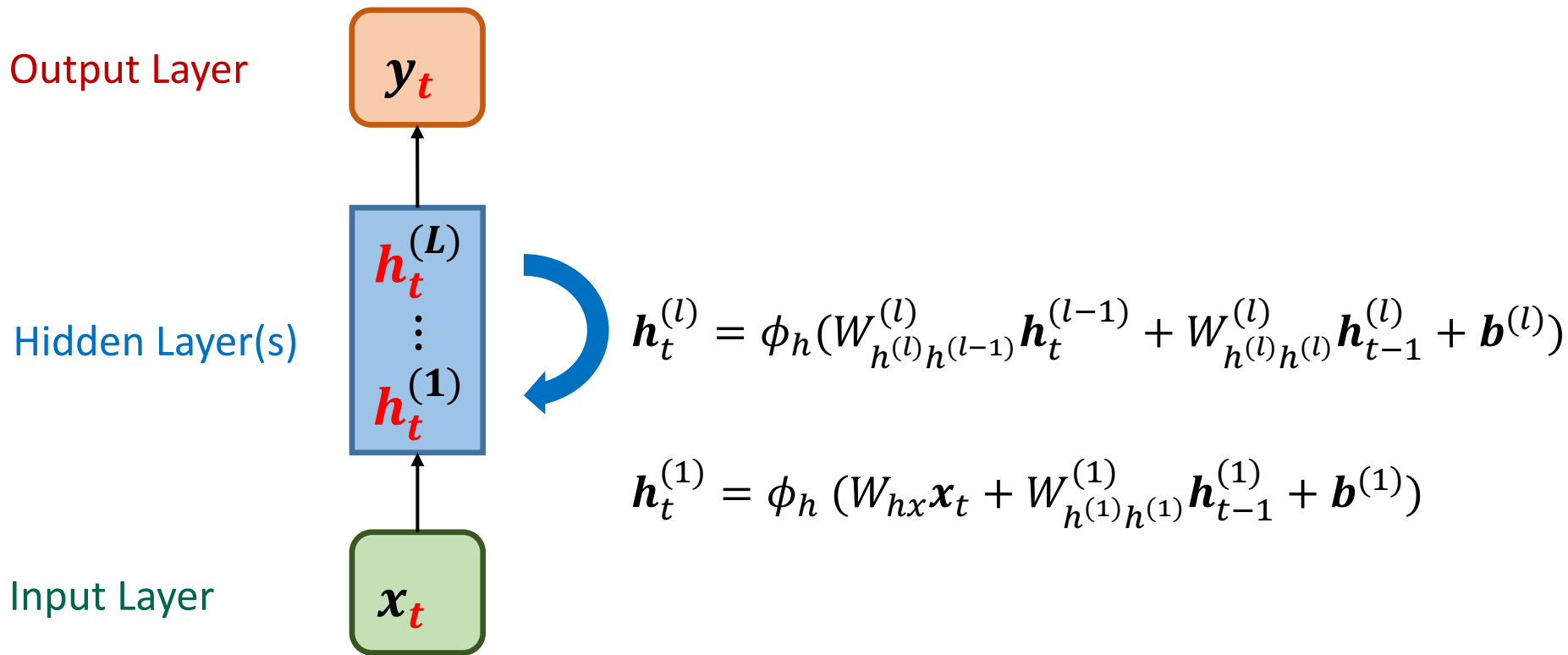
- Neural networks are able to represent non-linear functions, i.e.  $y_n \approx f(\mathbf{x}_n)$  can be non-linear
  - Neural networks are able to *learn* the features and the weights (parameters) from the data
  - Tutorial: <http://neuralnetworksanddeeplearning.com/>
-

# Recurrent Neural Network

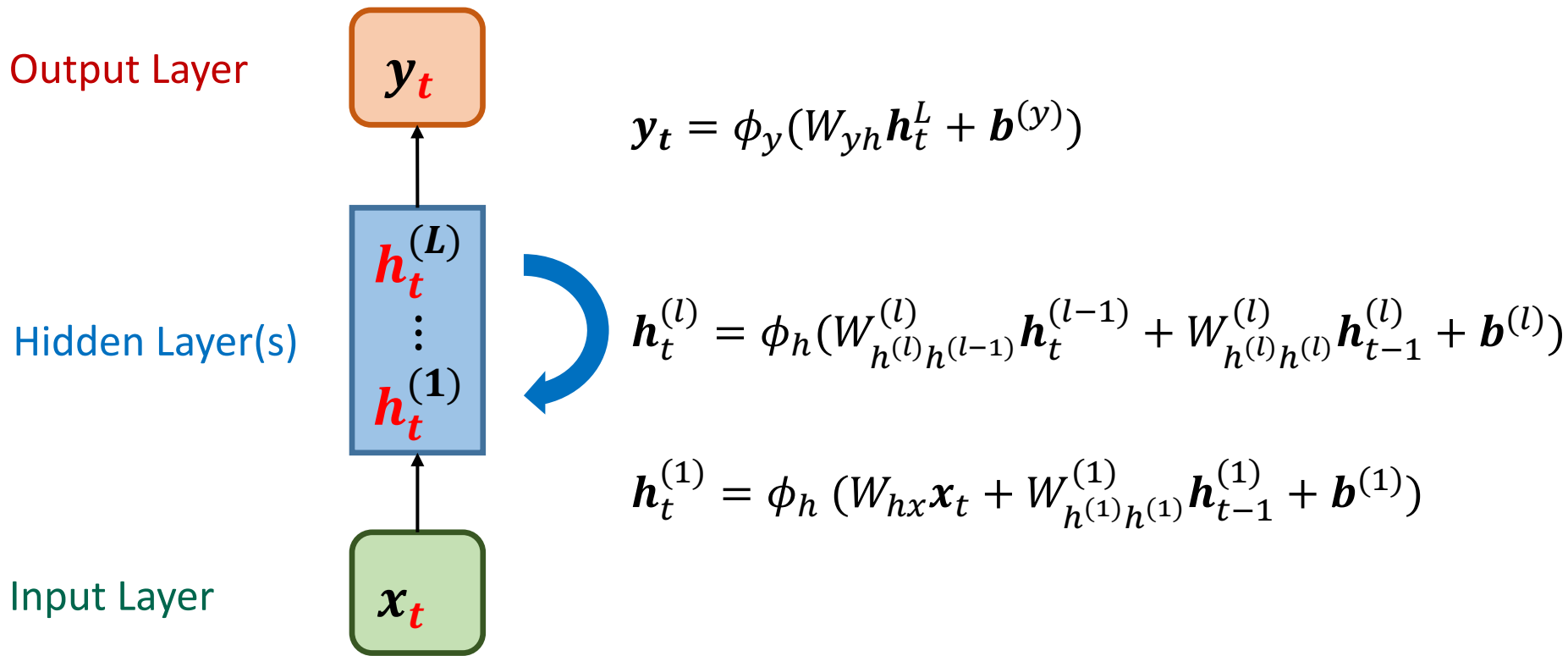


$$\mathbf{h}_t^{(1)} = \phi_h (W_{hx}\mathbf{x}_t + W_{h^{(1)}h^{(1)}}^{(1)}\mathbf{h}_{t-1}^{(1)} + \mathbf{b}^{(1)})$$

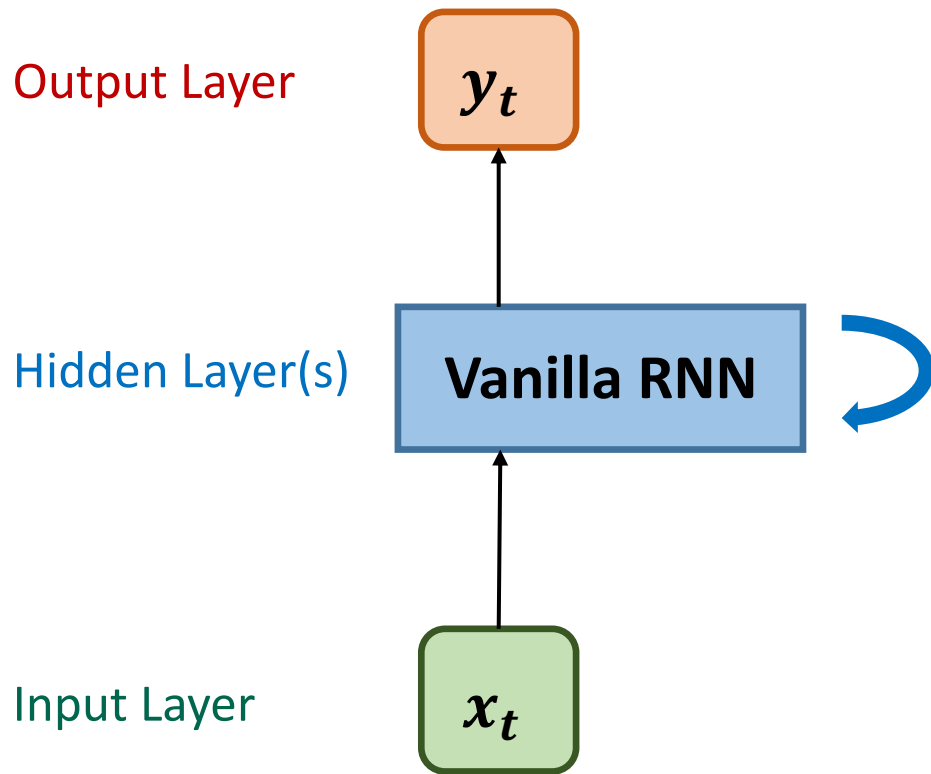
# Recurrent Neural Network



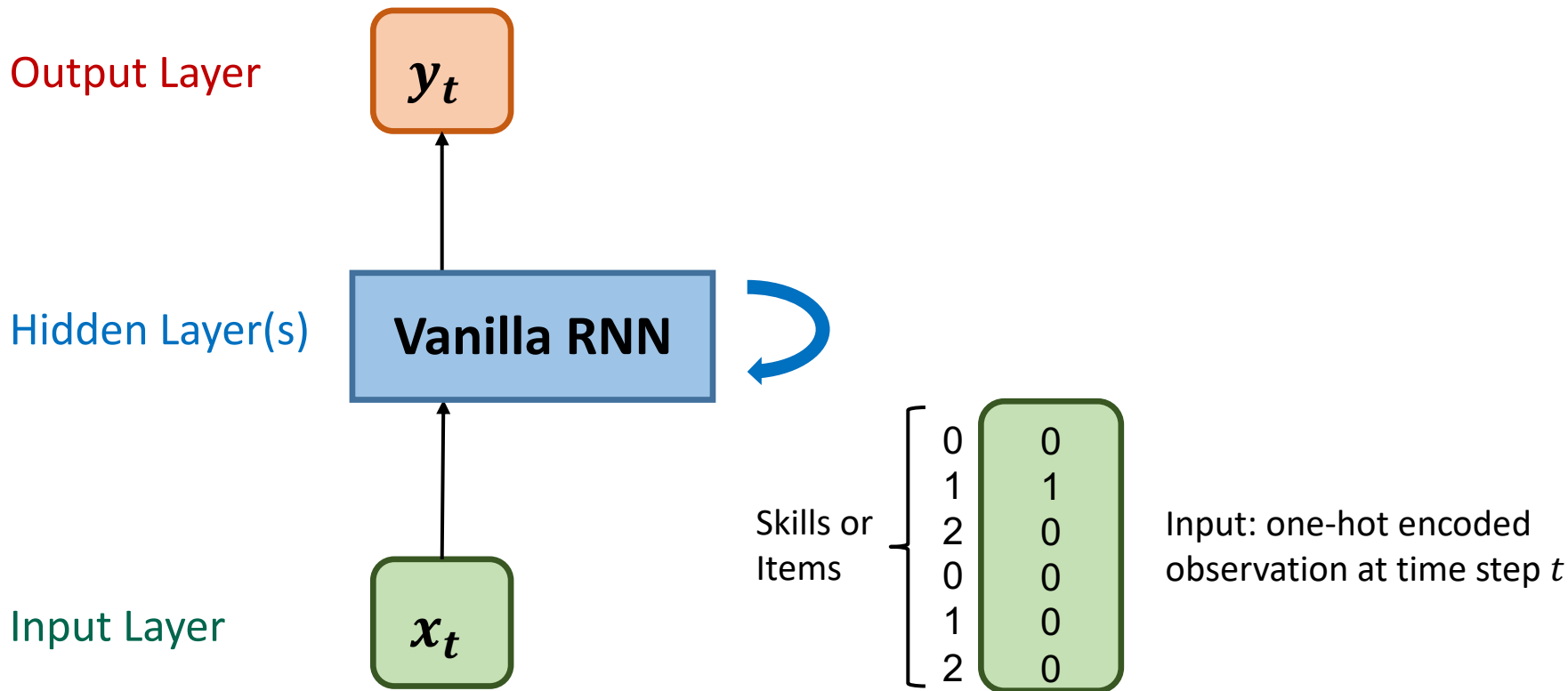
# Recurrent Neural Network



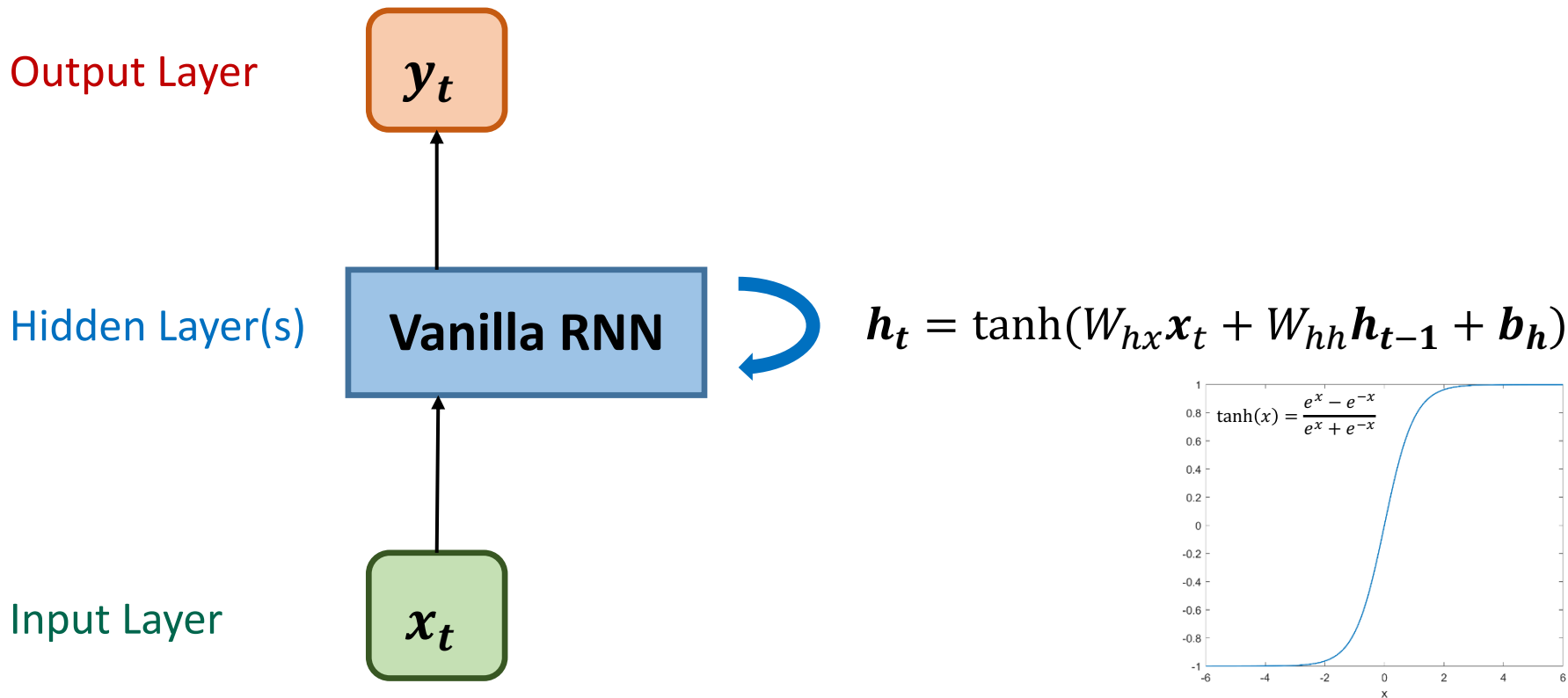
# Deep Knowledge Tracing



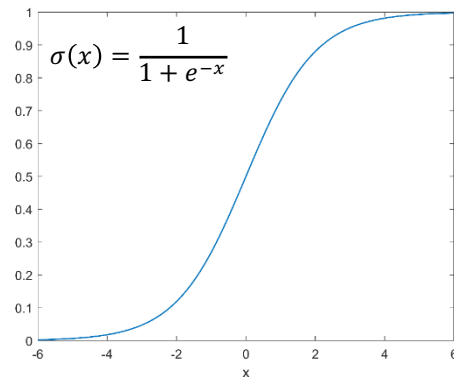
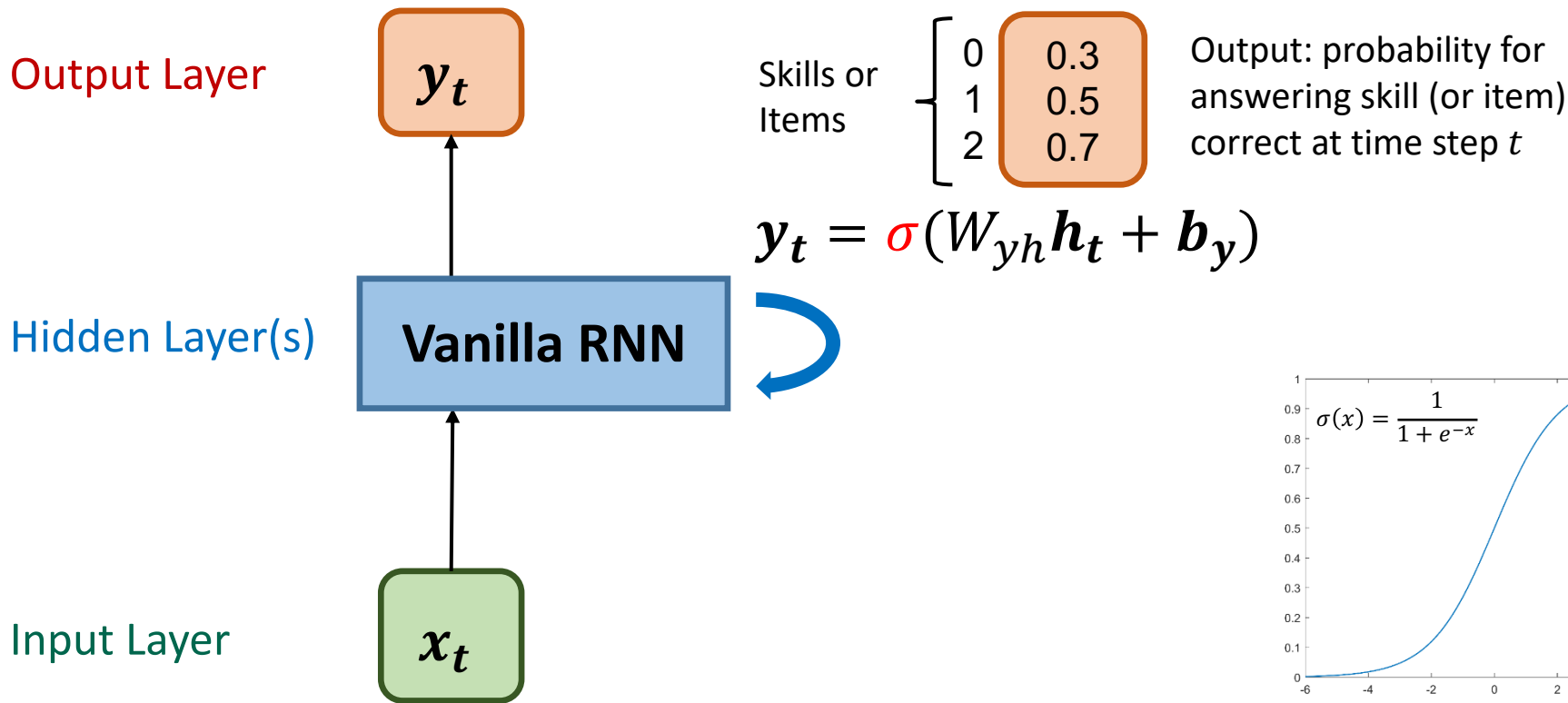
# Deep Knowledge Tracing



# Deep Knowledge Tracing

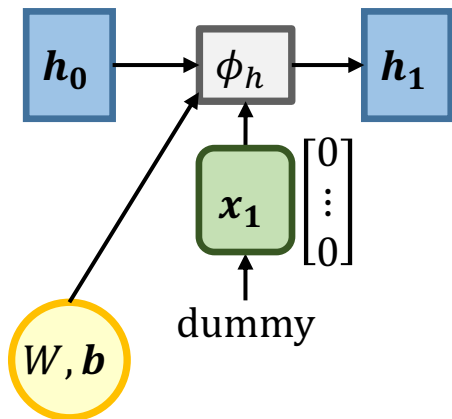


# Deep Knowledge Tracing

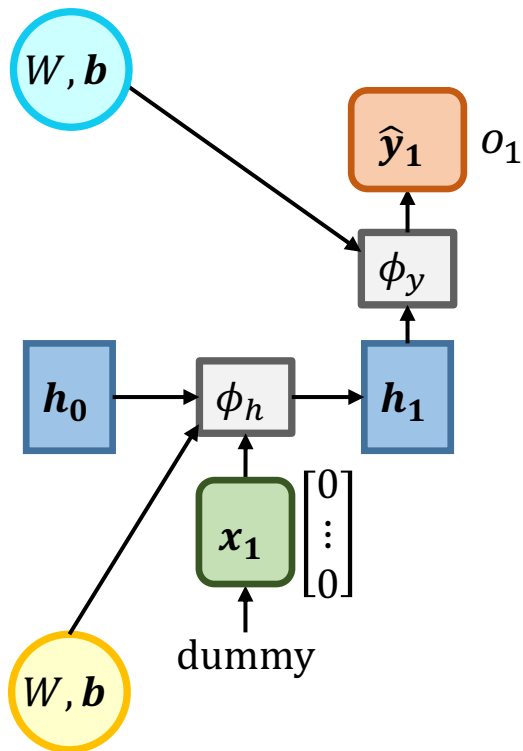




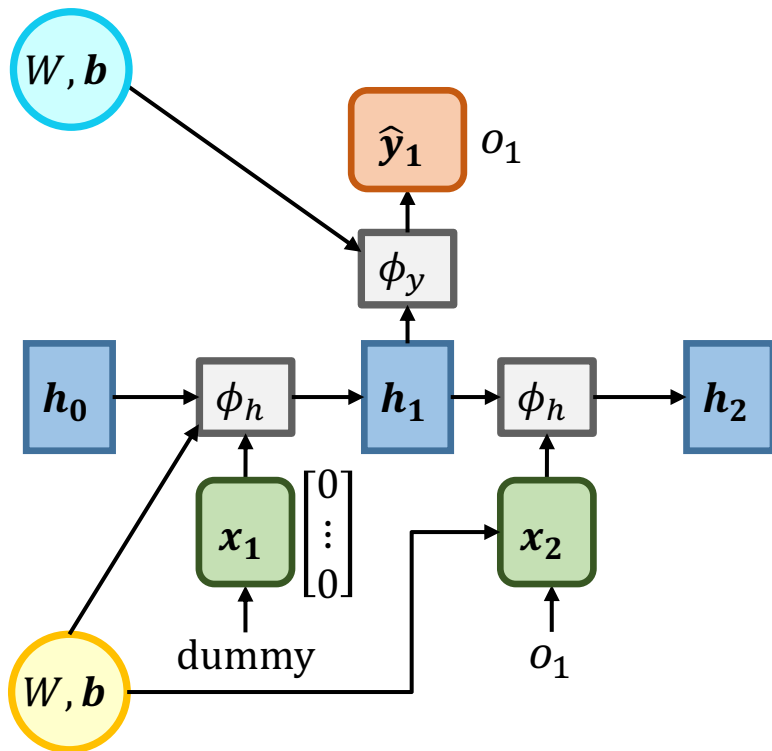
# Deep Knowledge Tracing – Computational Graph



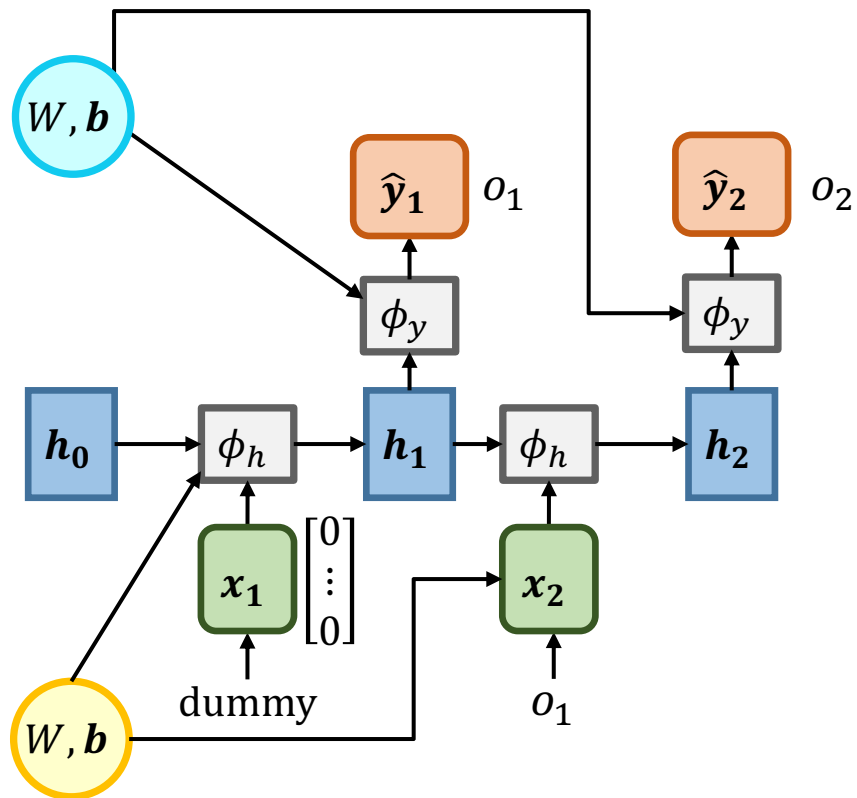
# Deep Knowledge Tracing – Computational Graph



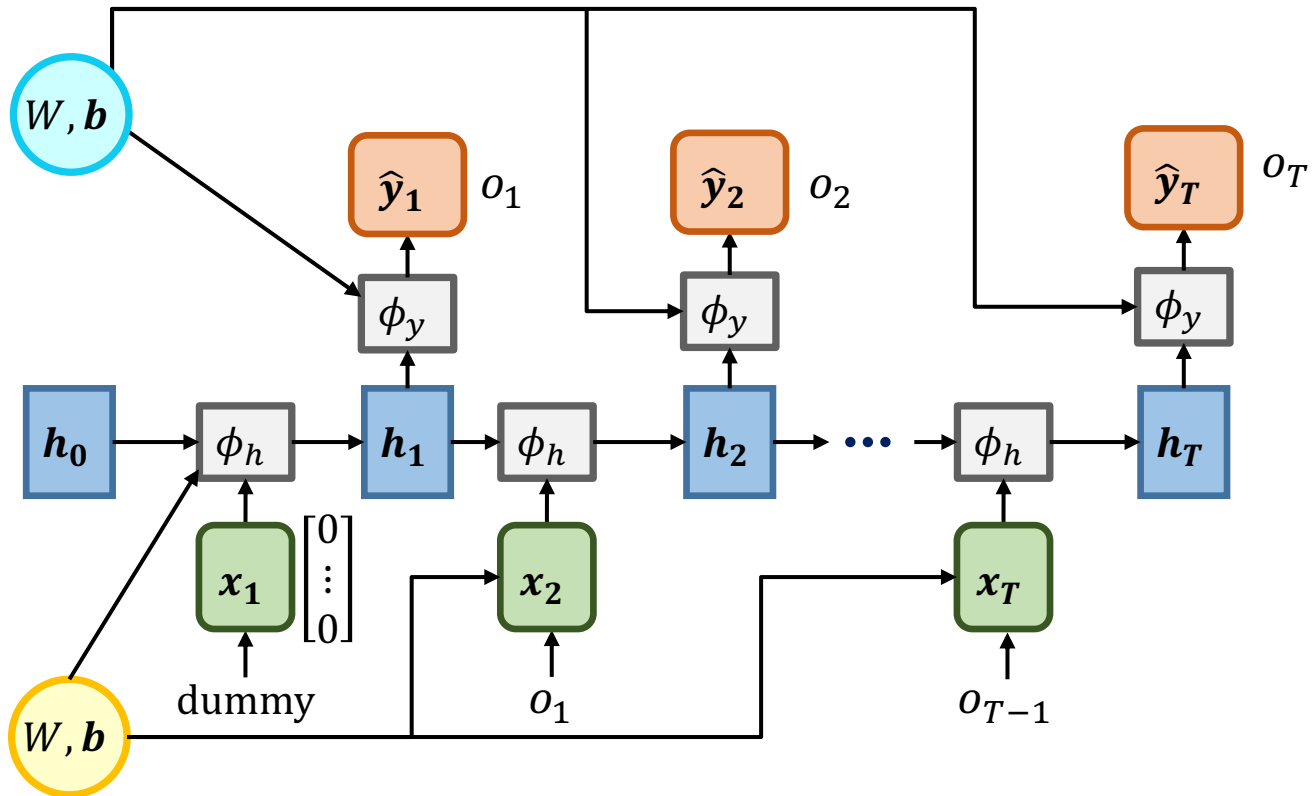
# Deep Knowledge Tracing – Computational Graph



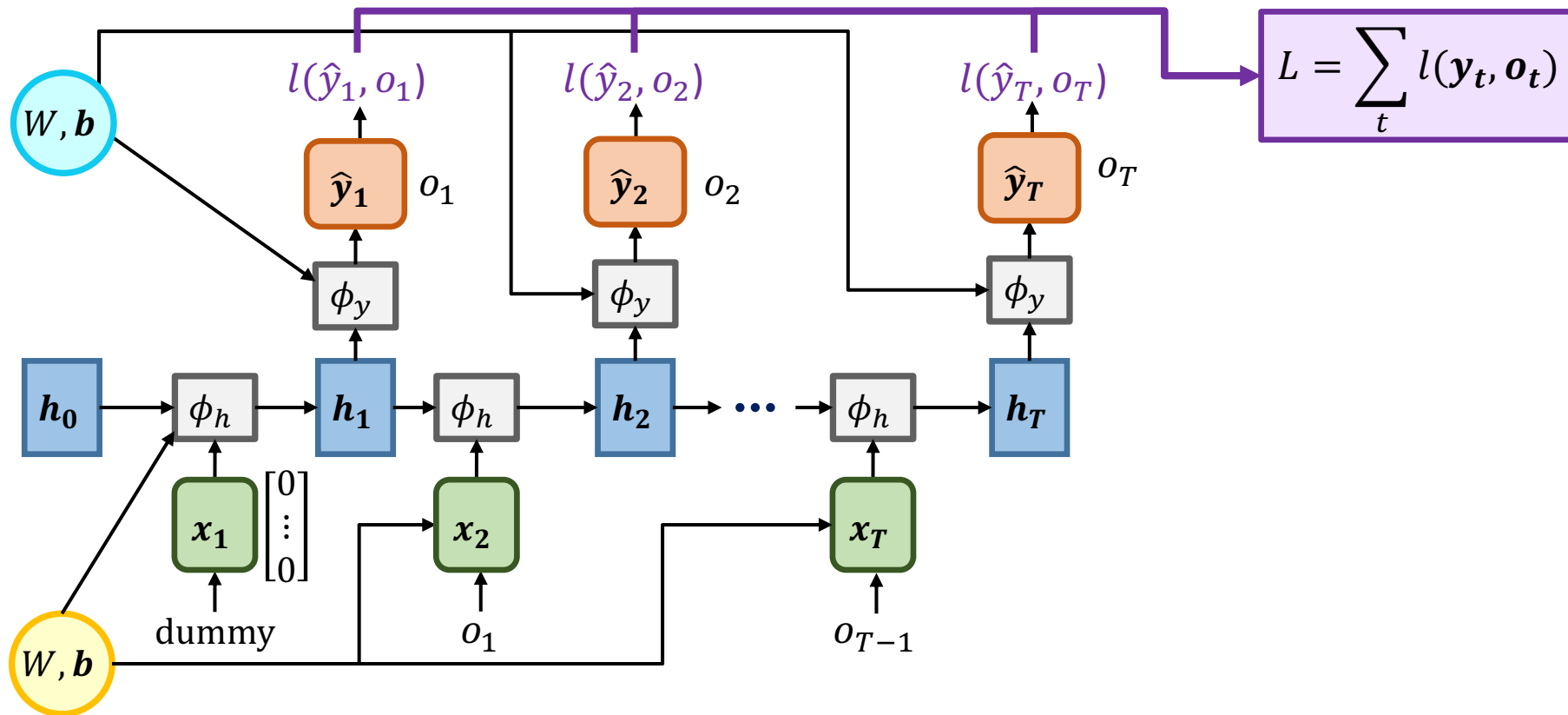
# Deep Knowledge Tracing – Computational Graph



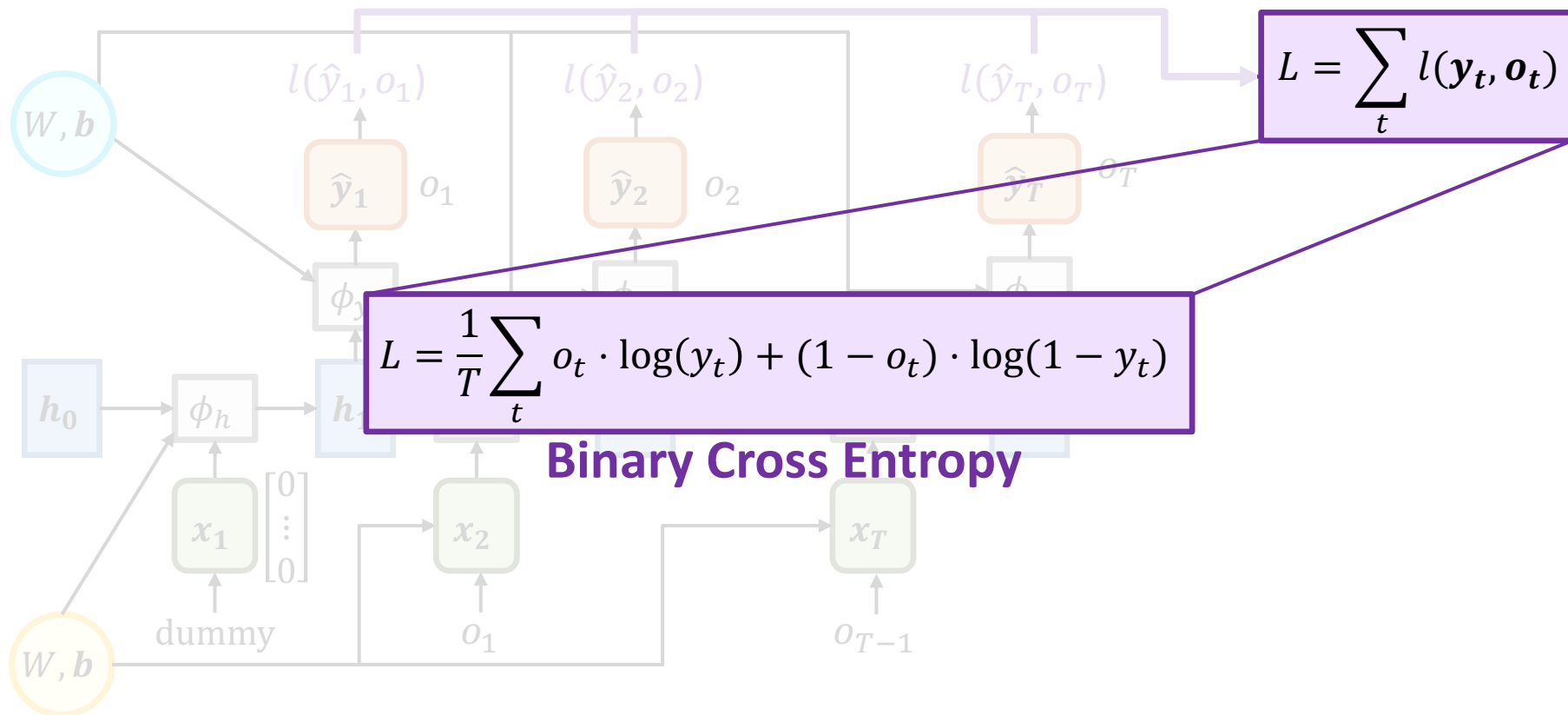
# Deep Knowledge Tracing – Computational Graph



# Deep Knowledge Tracing – Computational Graph



# Training a DKT model: Binary Crossentropy Loss



# Training and Prediction using DKT

- Training: gradient descent
  - Prediction: compute inference in the network (see computational graph)
-



# Today – Recurrent Neural Networks

- Deep Knowledge Tracing
  - **Parameters and hyperparameter tuning**
  - Different architectures
  - Different tasks:
    - “Many-to-many” versus “Many-to-one”
    - Classification versus Regression
-

# RNNs – Specifying Parameters

```
[ ] # Specify the model hyperparameters. Full descriptions included in the demo notebook!  
params = {}  
  
params['batch_size'] = 32  
params['mask_value'] = -1.0  
params['verbose'] = 1  
params['best_model_weights'] = 'weights/bestmodel'  
params['optimizer'] = 'adam'  
params['recurrent_units'] = 16  
params['epochs'] = 20  
params['dropout_rate'] = 0.1
```

# RNNs – Tuning hyperparameters

```
[ ] # Specify the model hyperparameters. Full descriptions included in the demo notebook!  
params = {}  
  
params['batch_size'] = 32  
params['mask_value'] = -1.0  
params['verbose'] = 1  
params['best_model_weights'] = 'weights/bestmodel'  
params['optimizer'] = 'adam'  
params['recurrent_units'] = 16  
params['epochs'] = 20  
params['dropout_rate'] = 0.1
```

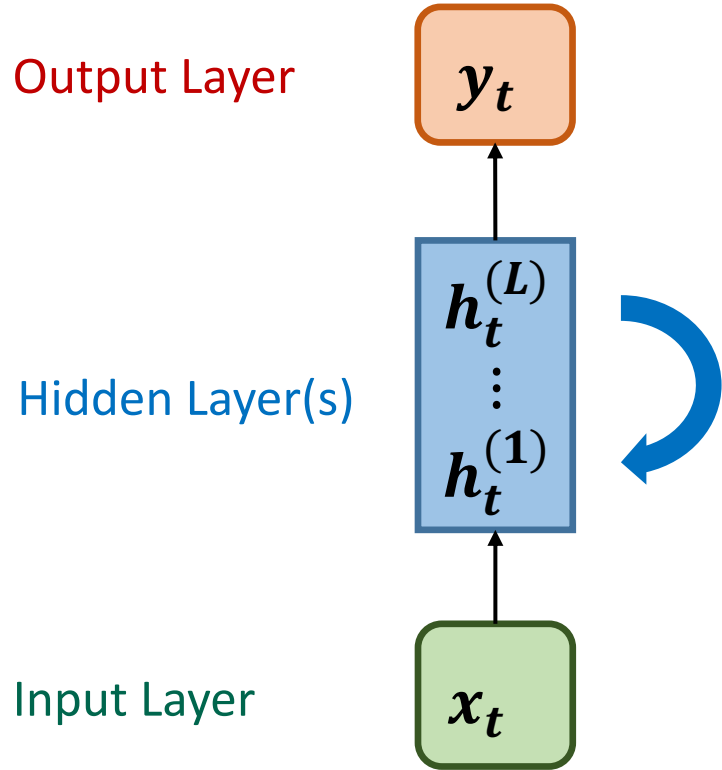
# RNNs – Tuning hyperparameters

- Optimal number of epochs can be found using callbacks
  - Other parameters can be tuned using for example:
    - a) Train-Validation-Test split
    - b) Train-Test split, using a k-fold cross validation on the training data to determine the optimal parameters
-

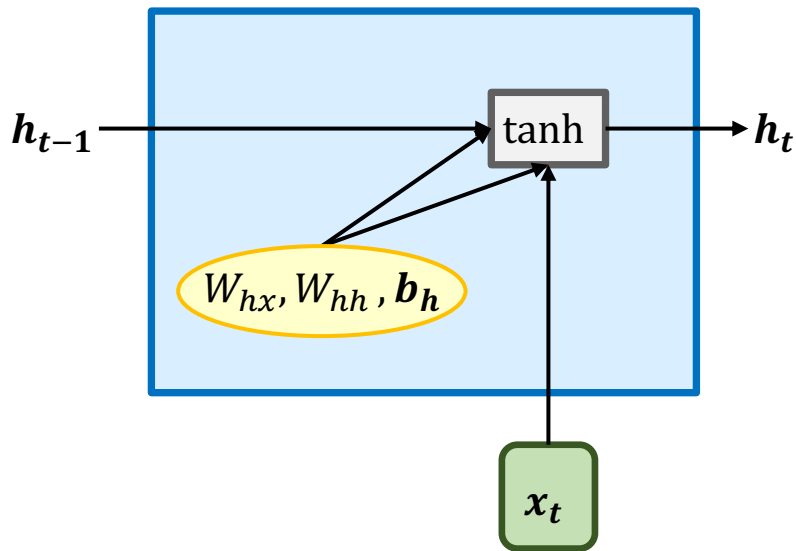
# Today – Recurrent Neural Networks

- Parameters and hyperparameter tuning
  - **Different architectures**
  - Different tasks:
    - “Many-to-many” versus “Many-to-one”
    - Classification versus Regression
-

# Recurrent Neural Network

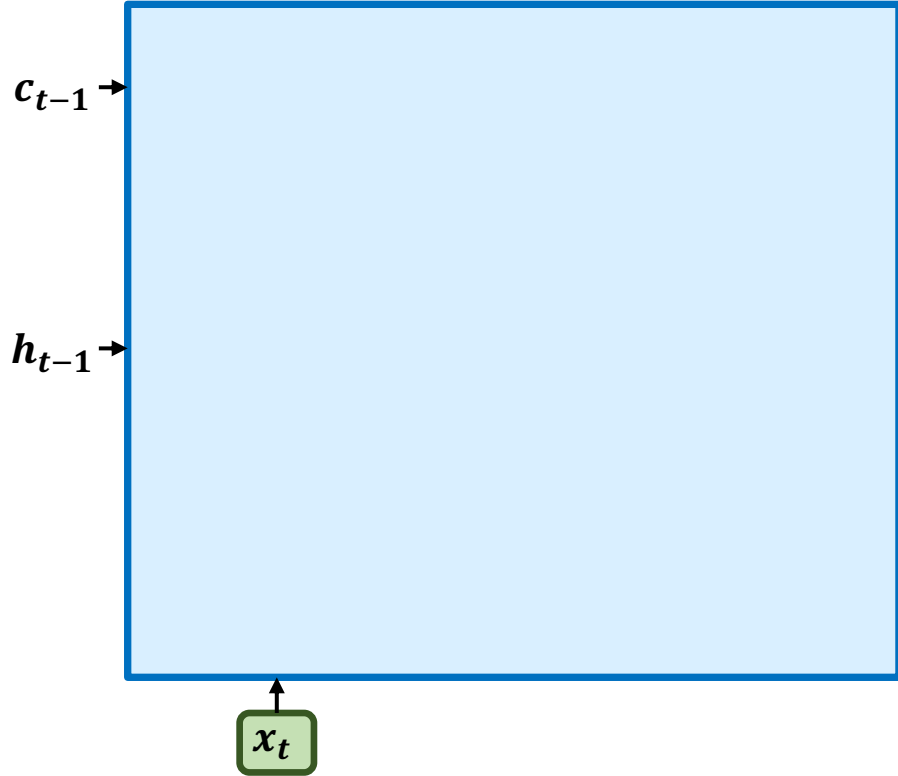


# Vanilla RNN - revisited



$$\mathbf{h}_t = \tanh(W_{hx}\mathbf{x}_t + W_{hh}\mathbf{h}_{t-1} + \mathbf{b}_h)$$

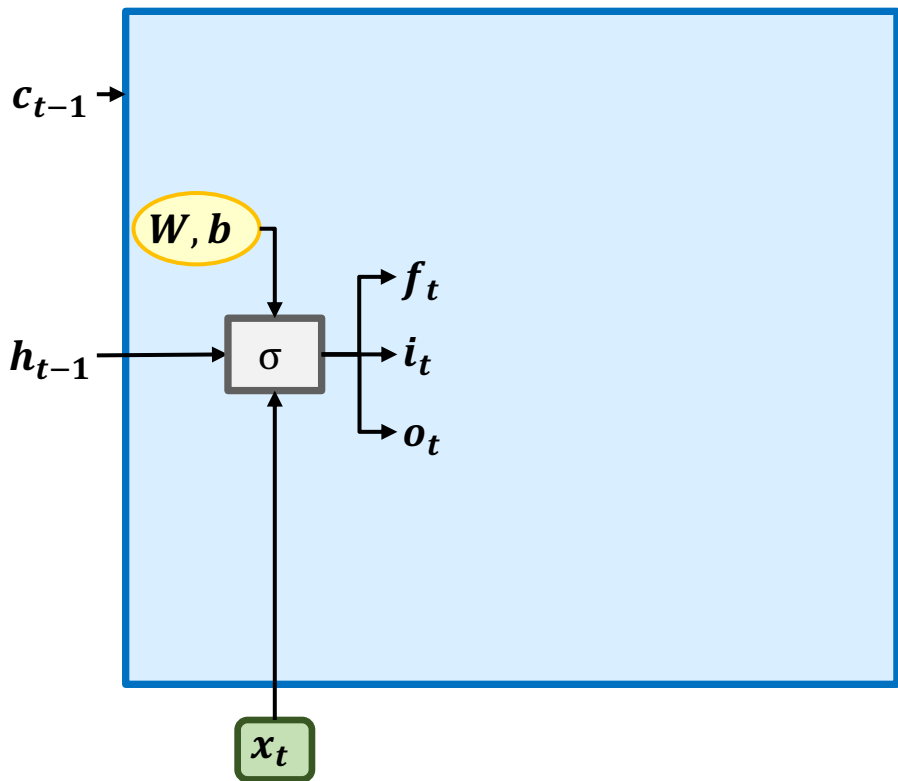
# Long-Short Term Memory Network (LSTM)



- Two states:
  - Hidden state  $h_{t-1}$
  - Cell state  $c_{t-1}$



# Long-Short Term Memory Network (LSTM)



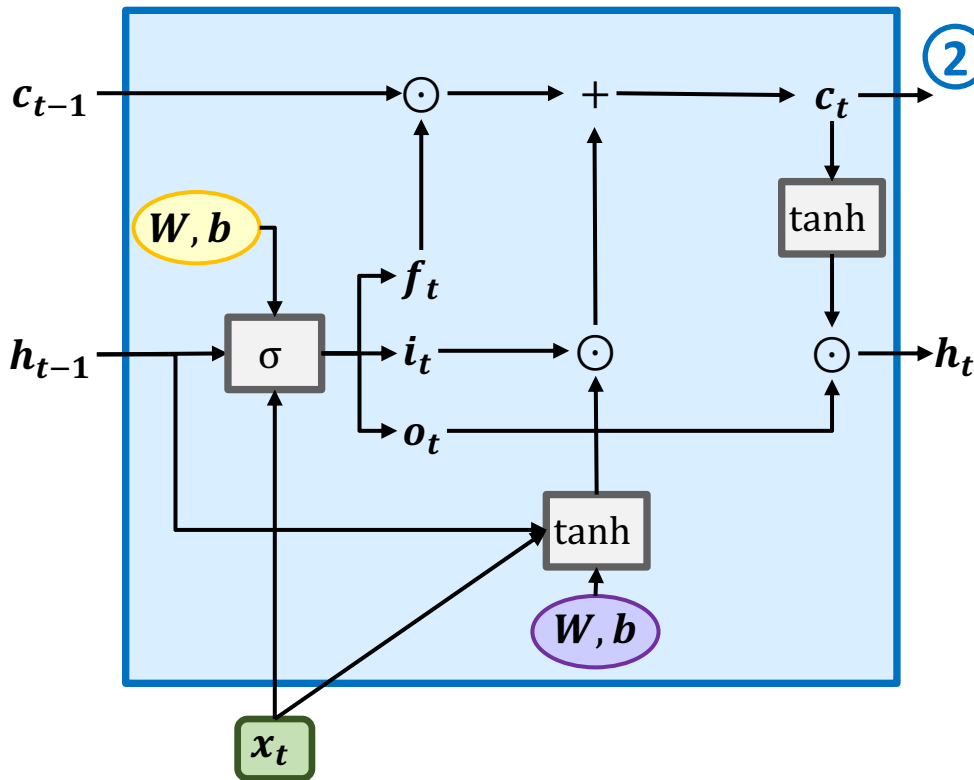
- ① Updating the gates:
- $f$  forget gate: whether to erase cell
  - $i$  input gate: whether to write to cell
  - $o$  output gate: how much to reveal cell

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f)$$

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i)$$

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o)$$

# Long-Short Term Memory Network (LSTM)

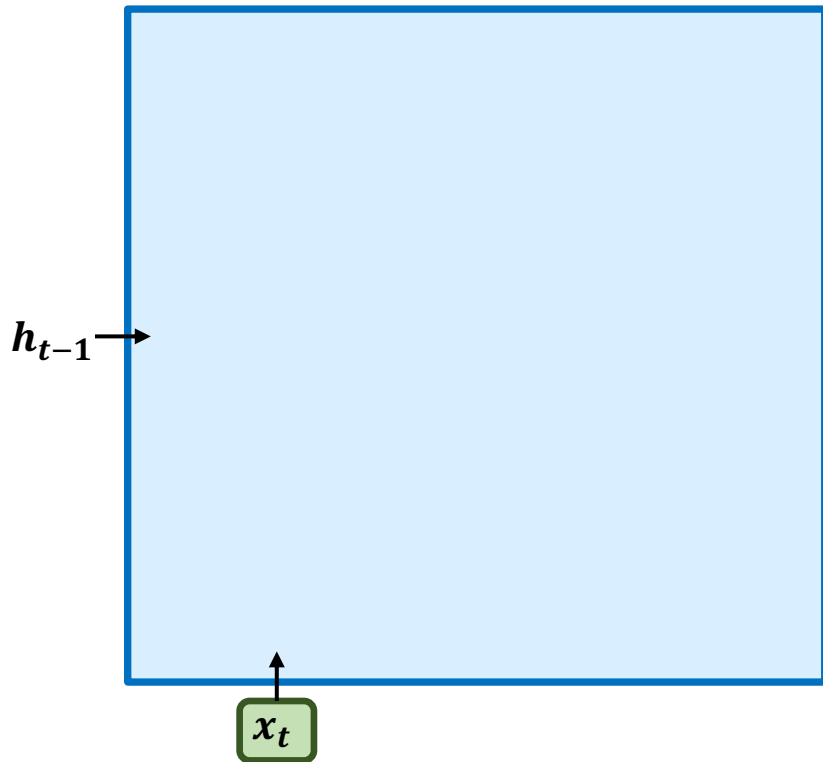


② Updating the states:

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_{cx}x_t + W_{ch}h_{t-1} + b_c)$$

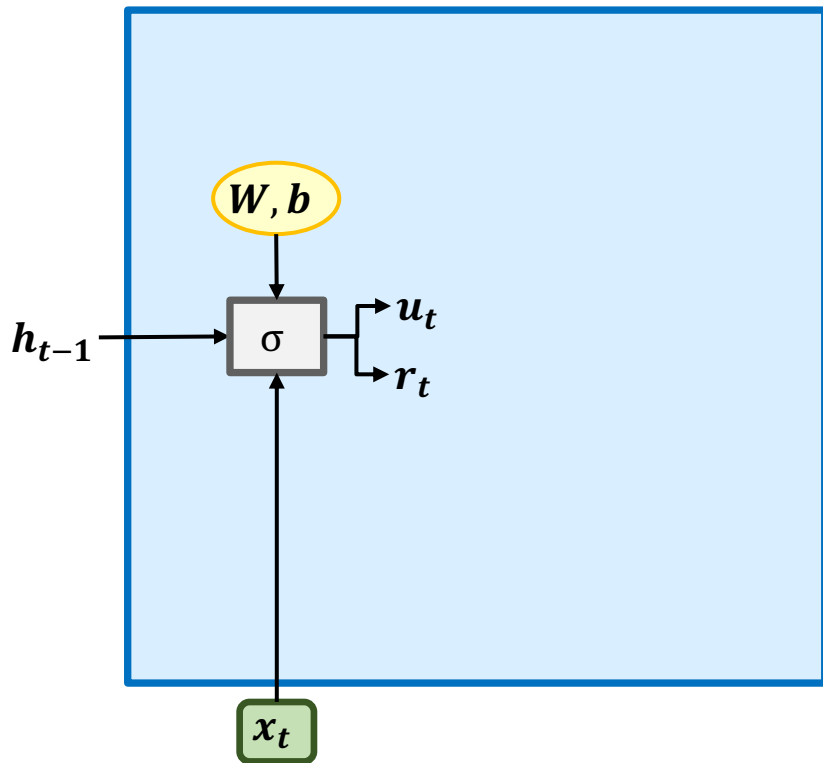
$$h_t = o_t \odot \tanh(c_t)$$

# Gated Recurrent Units (GRU)



- Only one state (got rid of cell):
  - Hidden state  $h_{t-1}$

# Gated Recurrent Units (GRU)

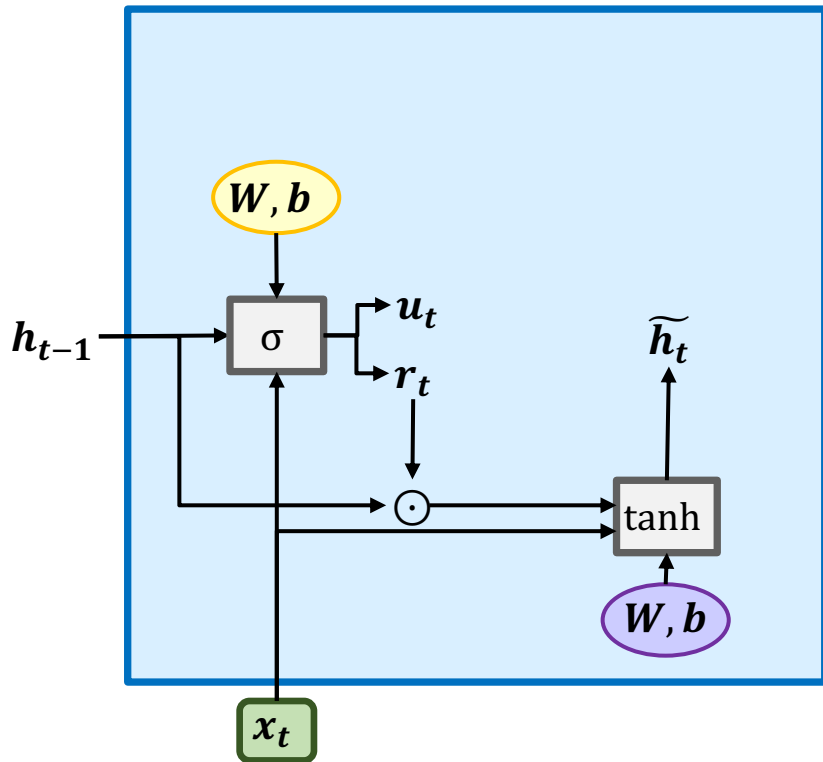


- ① Updating the gates:
- $r$  reset gate: how much of the previous state to remember
  - $u$  update gate: how much of the new state is just a copy of the old state

$$r_t = \sigma(W_{rx}x_t + W_{th}h_{t-1} + b_r)$$

$$u_t = \sigma(W_{ux}x_t + W_{uh}h_{t-1} + b_u)$$

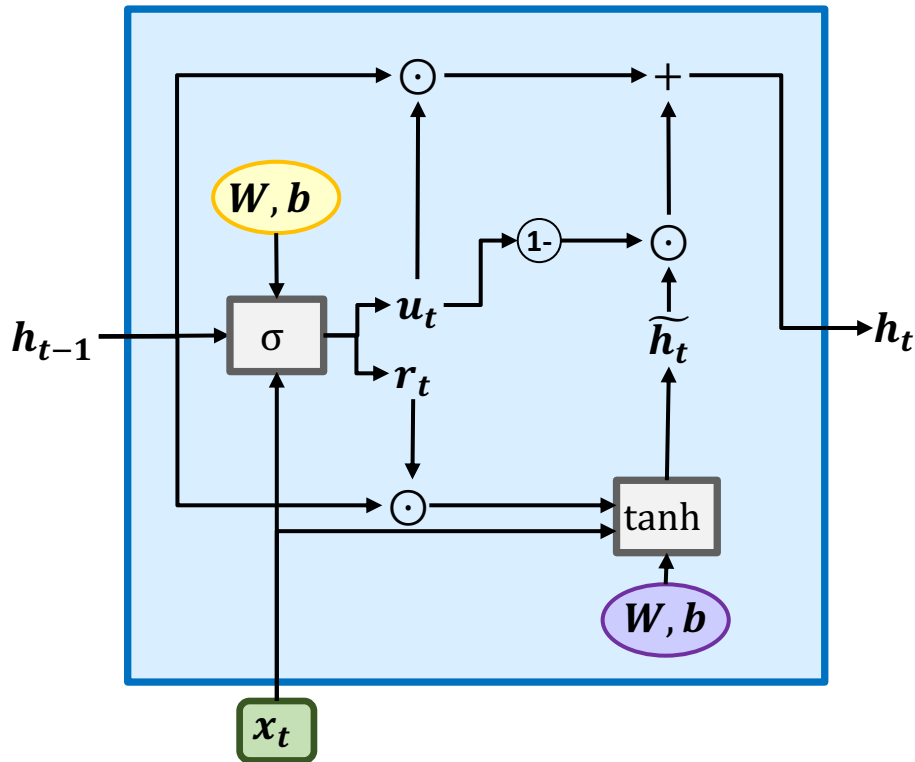
# Gated Recurrent Units (GRU)



② Get candidate hidden state:

$$\tilde{h}_t = \tanh(W_{hx}x_t + W_{ht}(r_t \odot h_{t-1}) + b_h)$$

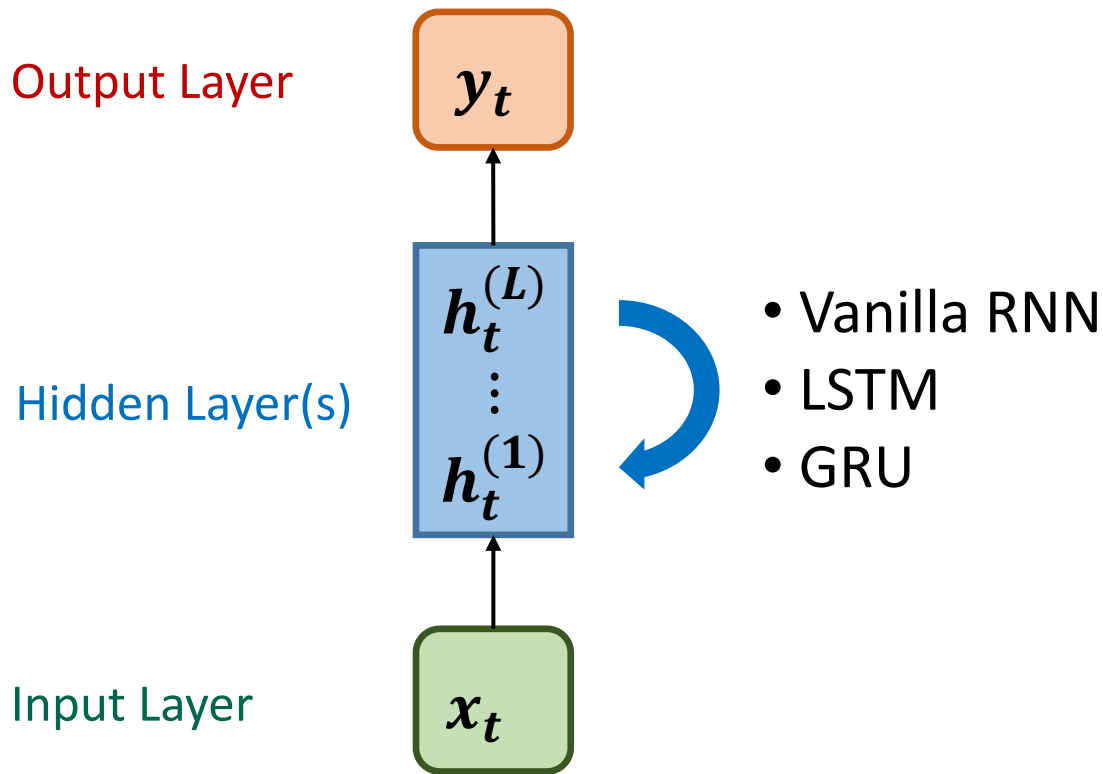
# Gated Recurrent Units (GRU)



③ Updating the state:

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

# Same input/output – different architectures



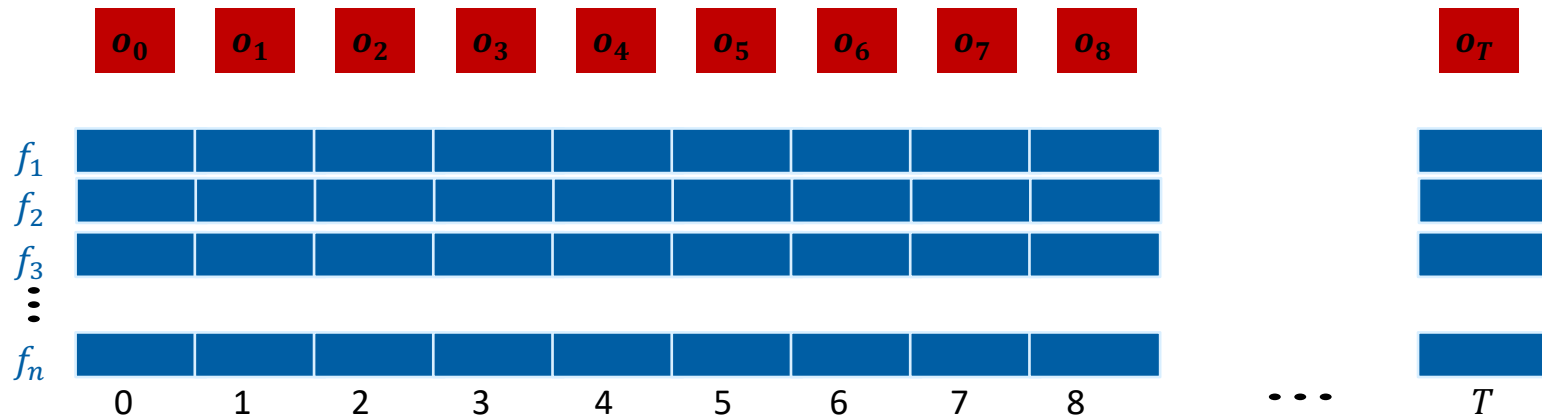
# Today – Recurrent Neural Networks

- Deep Knowledge Tracing
  - Parameters and hyperparameter tuning
  - Different architectures
  - **Different tasks:**
    - “Many-to-many” versus “Many-to-one”
    - Classification versus Regression
-

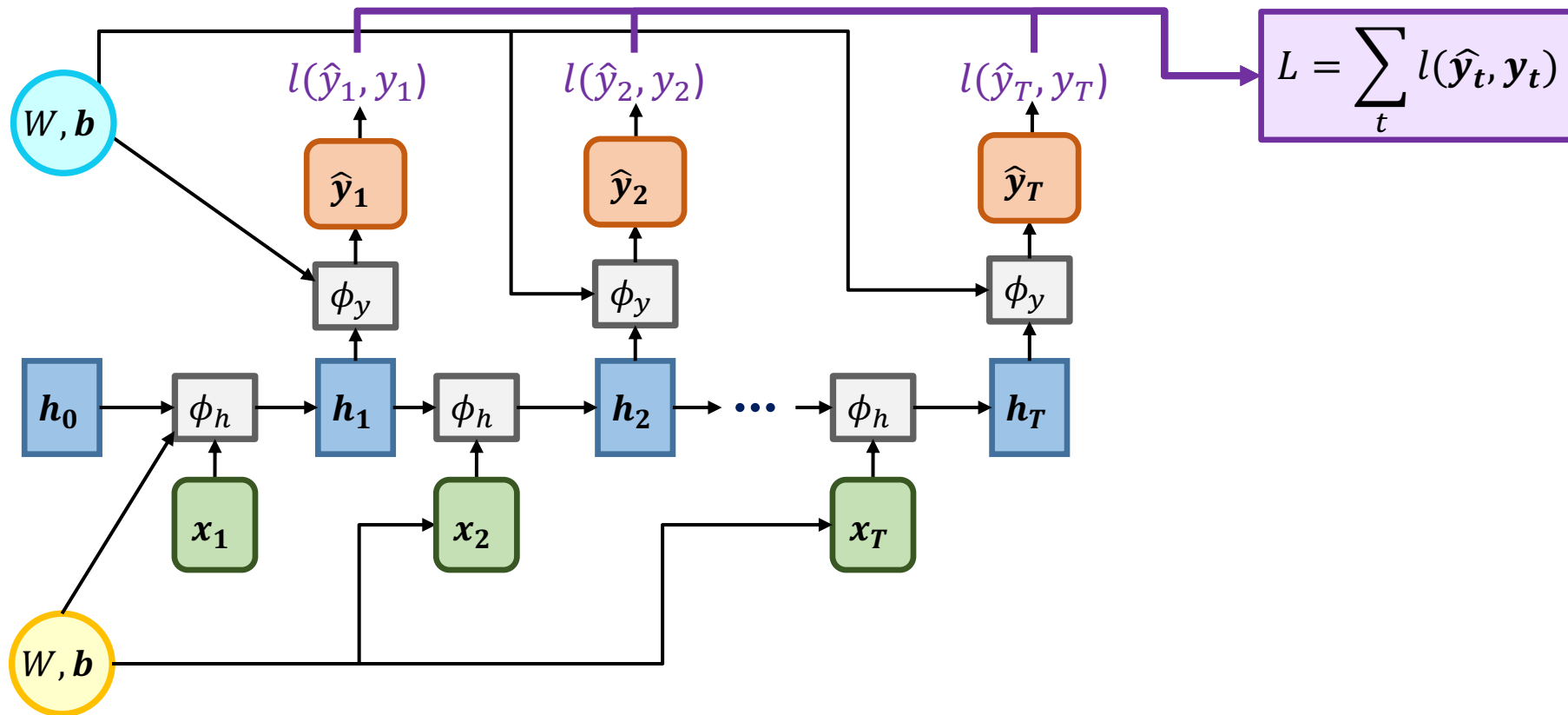


# Many-to-many aka the Tracing Task

- Prediction of a target variable  $o_t$  at each time step  $t$

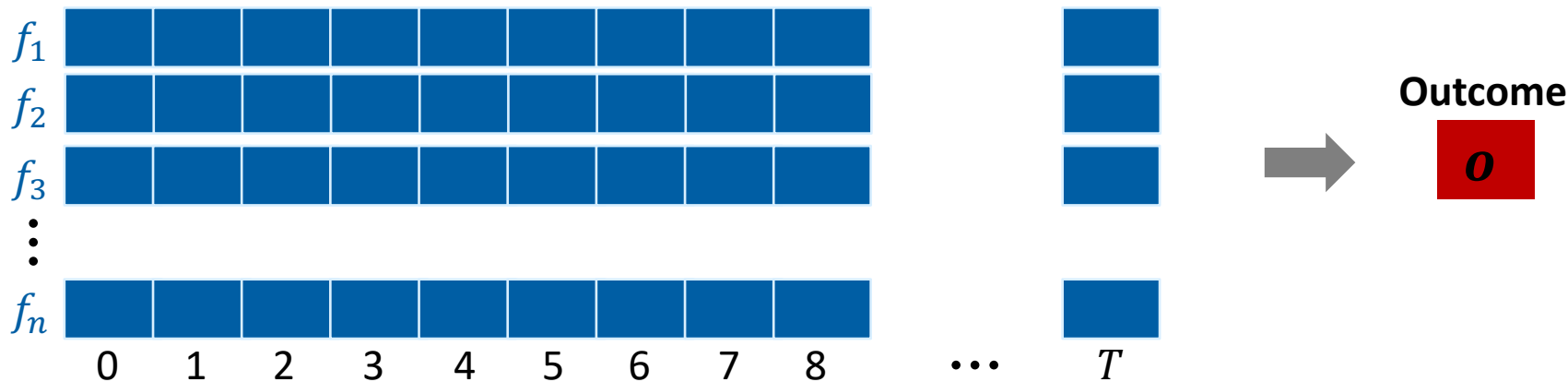


# Computational Graph – Many-to-many

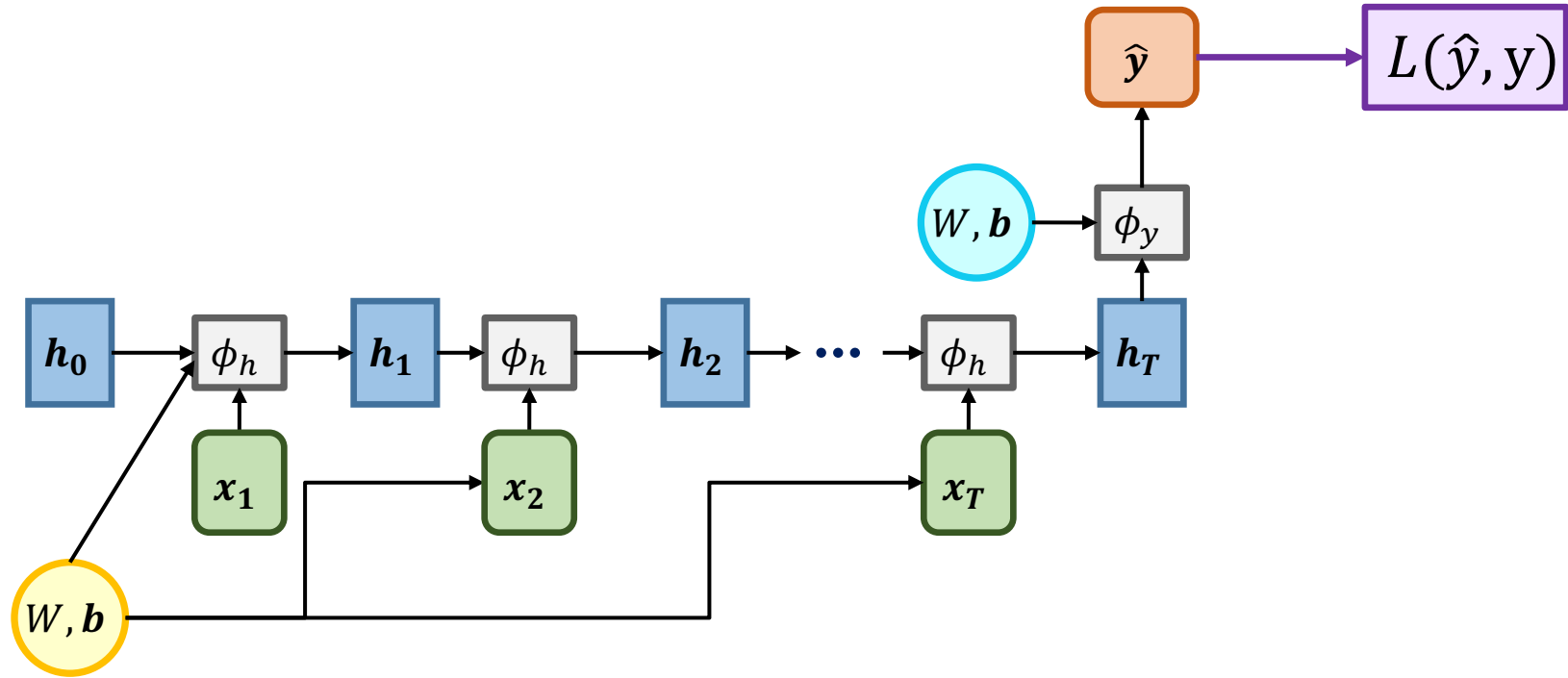


# Many-to-one aka the Time-Series Prediction Task

- Prediction of a target variable  $o$  after  $t \leq T$  time steps, where  $T$  is the total number of time steps



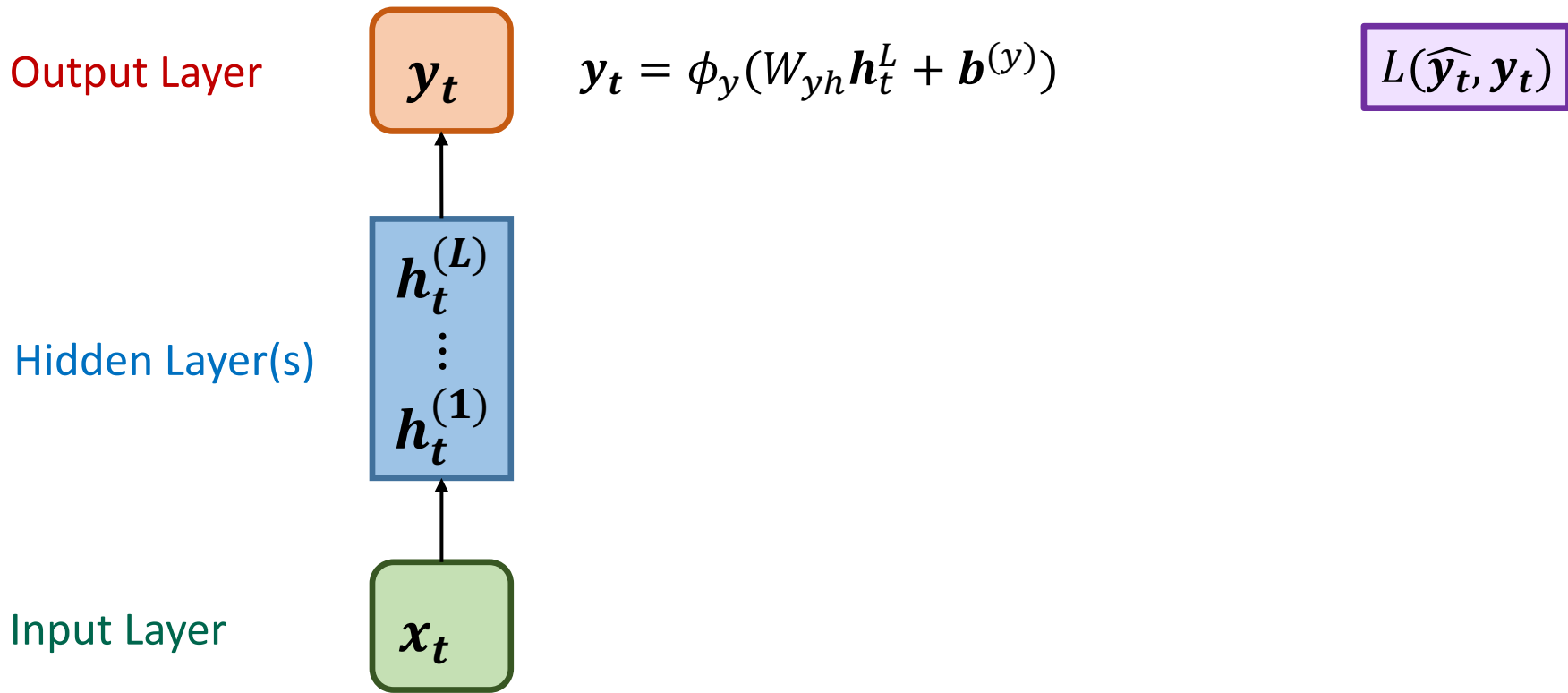
# Computational Graph – Many-to-one



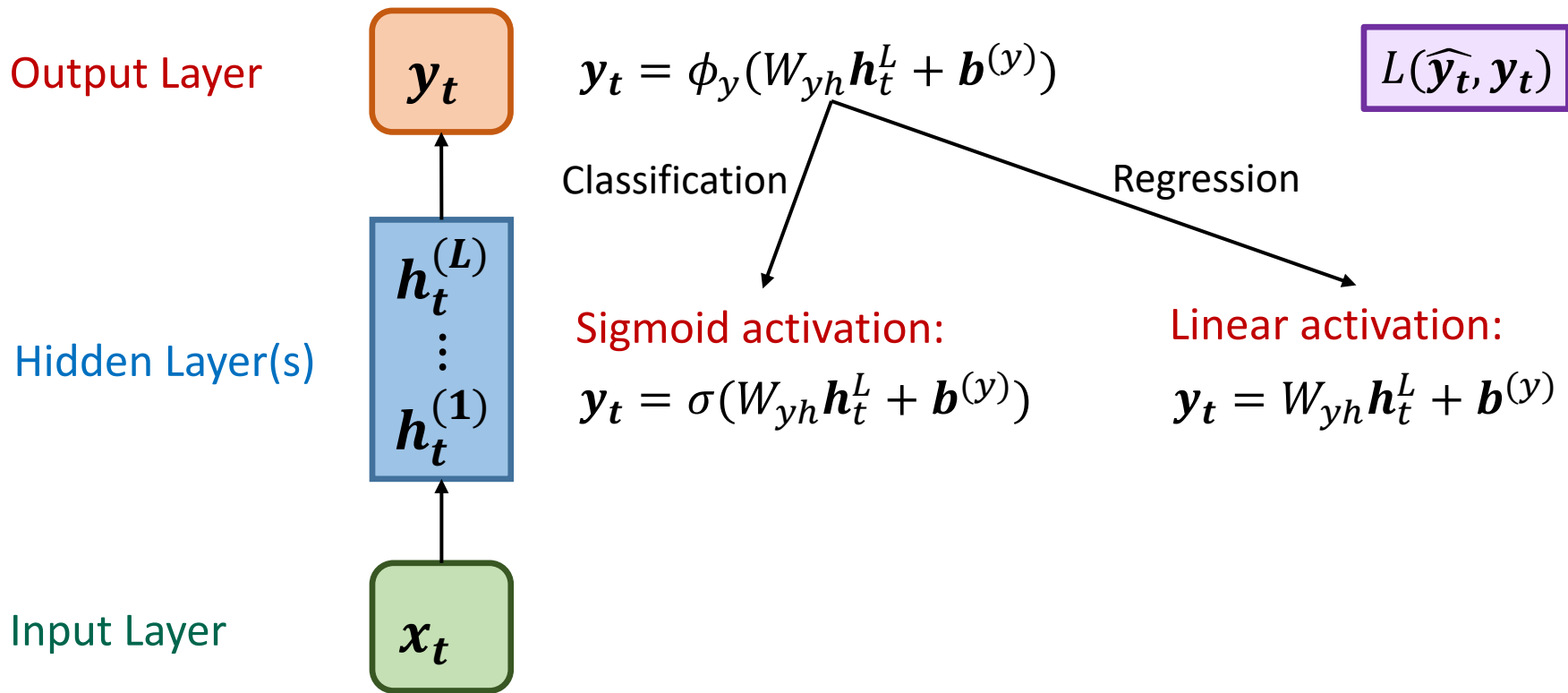
# Today – Recurrent Neural Networks

- Deep Knowledge Tracing
  - Parameters and hyperparameter tuning
  - Different architectures
  - **Different tasks:**
    - “Many-to-many” versus “Many-to-one”
    - **Classification versus Regression**
-

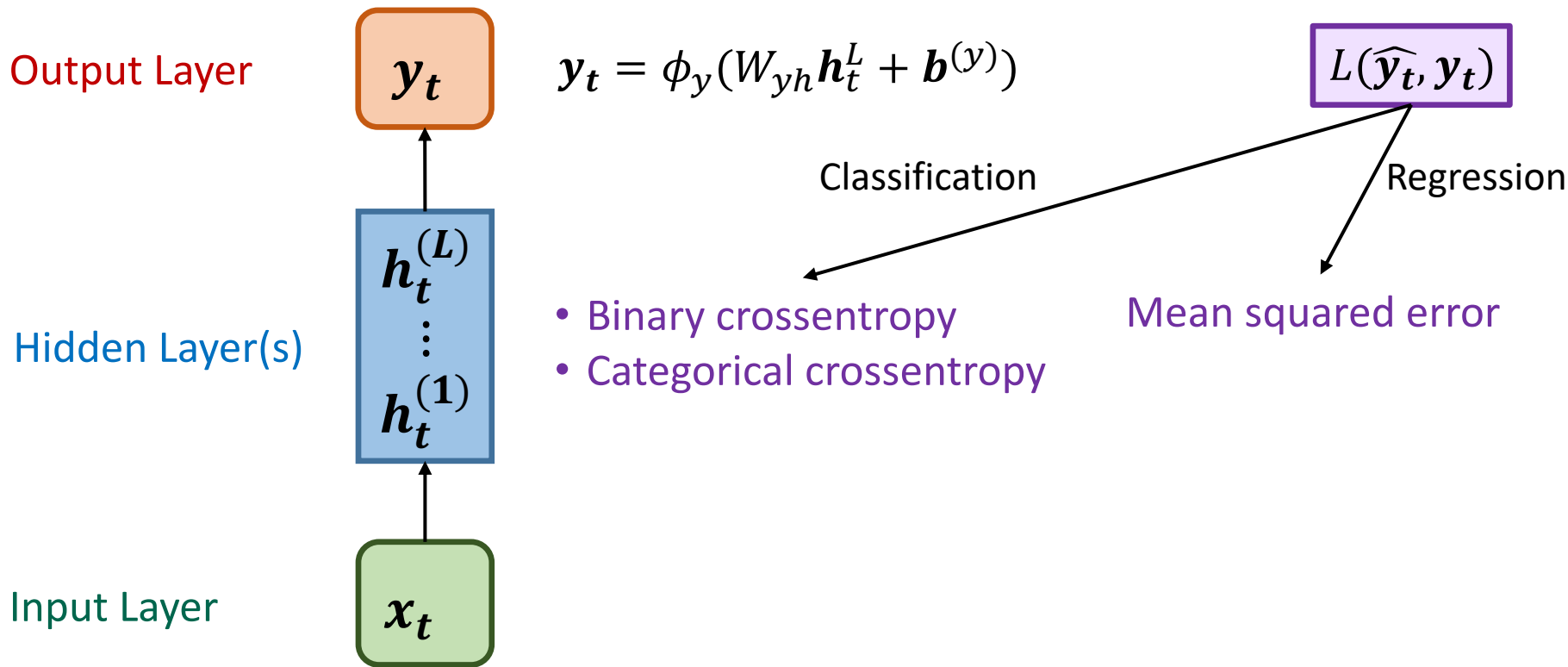
# Classification vs. Regression



# Classification vs. Regression: Output Layer



# Classification vs. Regression: Training Loss





# Your Turn

- Given:
    - Data from a MOOC
    - An LSTM for predicting quiz performance of a student for every week of the course (tracing task)
  - Your Task:
    - 1) Adjust the `create_model` function in order to predict pass/fail after 5 weeks of the course (time series prediction task) and send us the binary accuracy + AUC
      - Hint 1: `return_sequences=False`
      - Hint 2: what does `TimeDistributed(...)` do?
    - 2) Tune hyperparameters of your choice and send us binary accuracy and AUC
-

# Summary

- Deep Knowledge Tracing
  - Parameters and hyperparameter tuning
  - Different architectures
  - Different tasks:
    - “Many-to-many” versus “Many-to-one”
    - Classification versus Regression
-