E9: 309 Advanced Deep Learning

7-10-2020

Instructor: Sriram Ganapathy
sriramg@iisc.ac.in

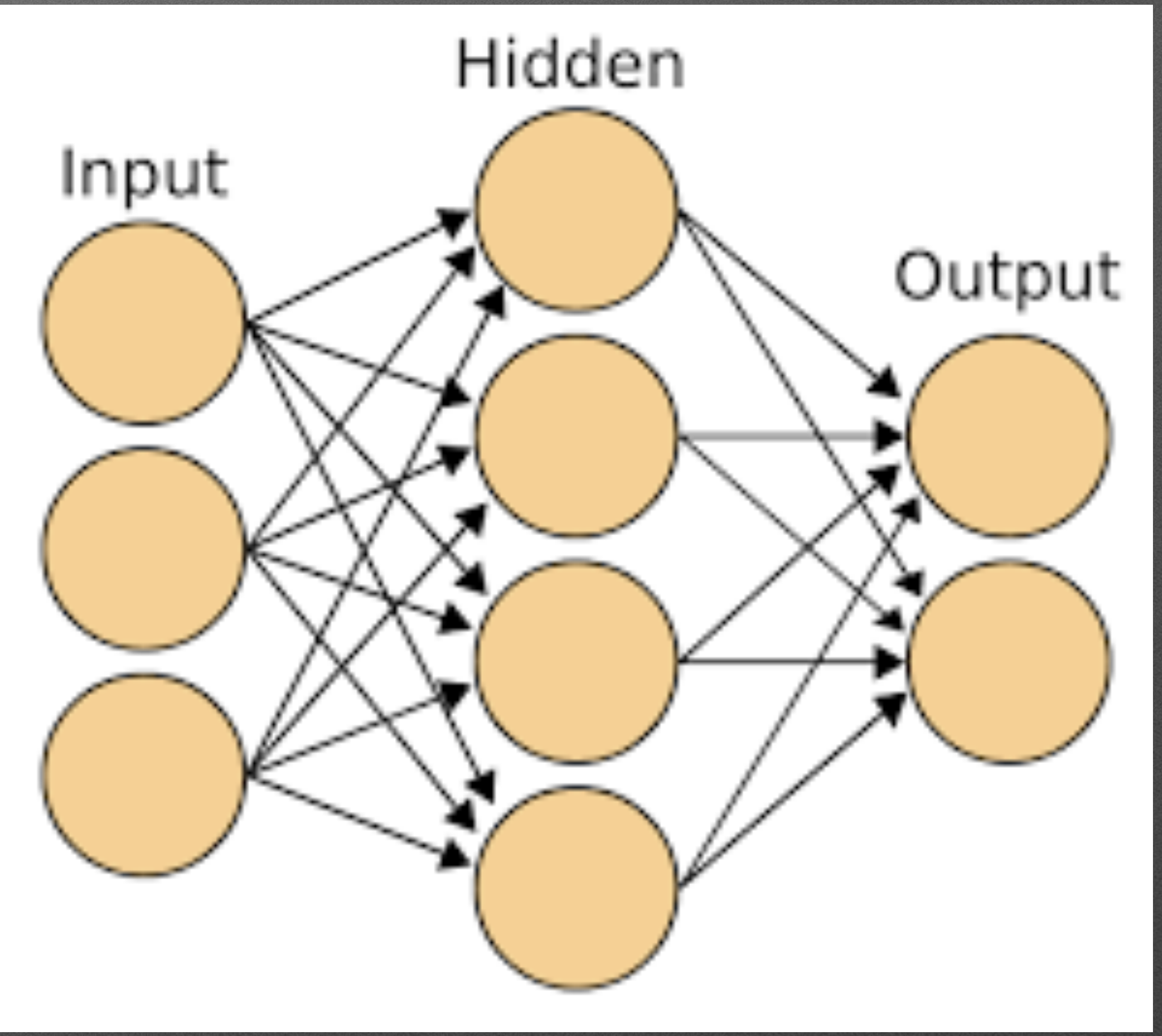
Teaching Assistant : Jaswanth Reddy
jaswanthk@iisc.ac.in

Schedule - MW - 330-5pm (Microsoft Teams)
http://leap.ee.iisc.ac.in/sriram/teaching/ADL2020/

Recap of deep learning

Some notations

- $\boldsymbol{x} \in \mathcal{R}^D$ - input data.
- $\boldsymbol{y} \in \mathcal{R}^C$ - neural network targets.
- $\hat{\boldsymbol{y}} \in \mathcal{B}^C$ - model outputs.
- $\boldsymbol{e}, \boldsymbol{h} \in \mathcal{R}^d$ - hidden model representations or embeddings.
- $\boldsymbol{\Theta}$ - collection of learnable parameters in the model.
- $E(\boldsymbol{y}, \hat{\boldsymbol{y}})$ - error function used in the model training.



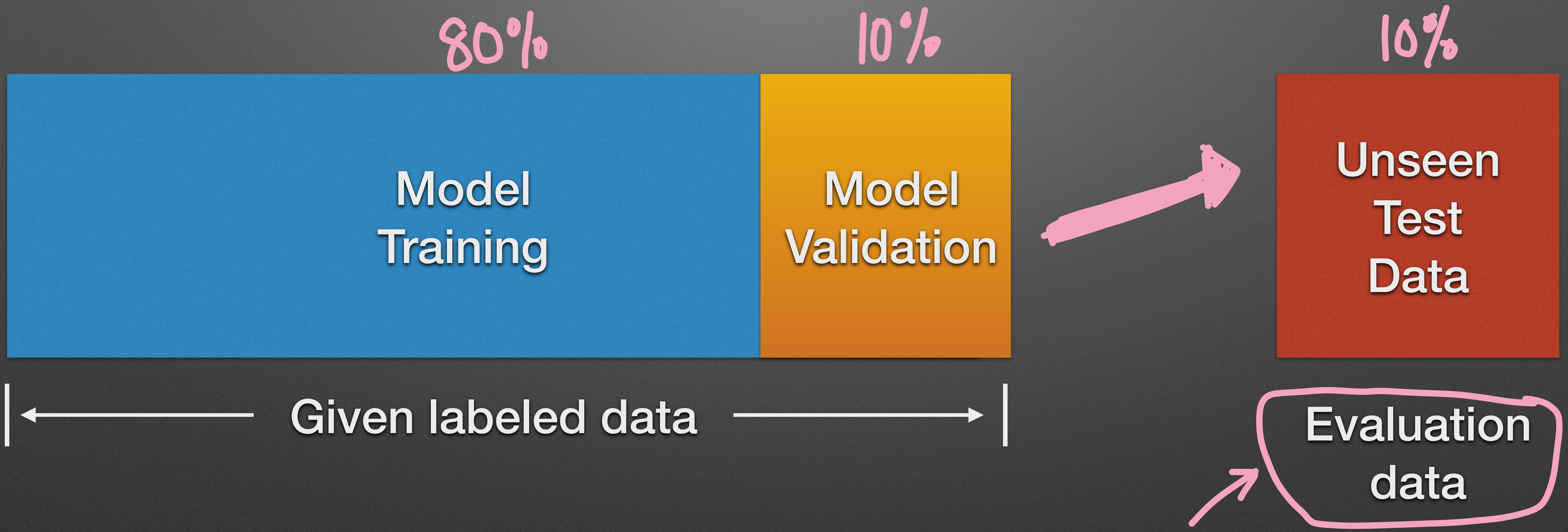
Some notations

- $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_N, \boldsymbol{y}_1, ..., \boldsymbol{y}_N\}$ - labeled training data
- $q = \{1...Q\}$ - iteration index. epoch.
- $t = \{1...T\}$ - discrete time index.
- $l = \{1...L\}$ - layer index
- η - learning rate (hyper-parameter)
- N_b - mini-batch size and B is the number of mini-batches.

Premise

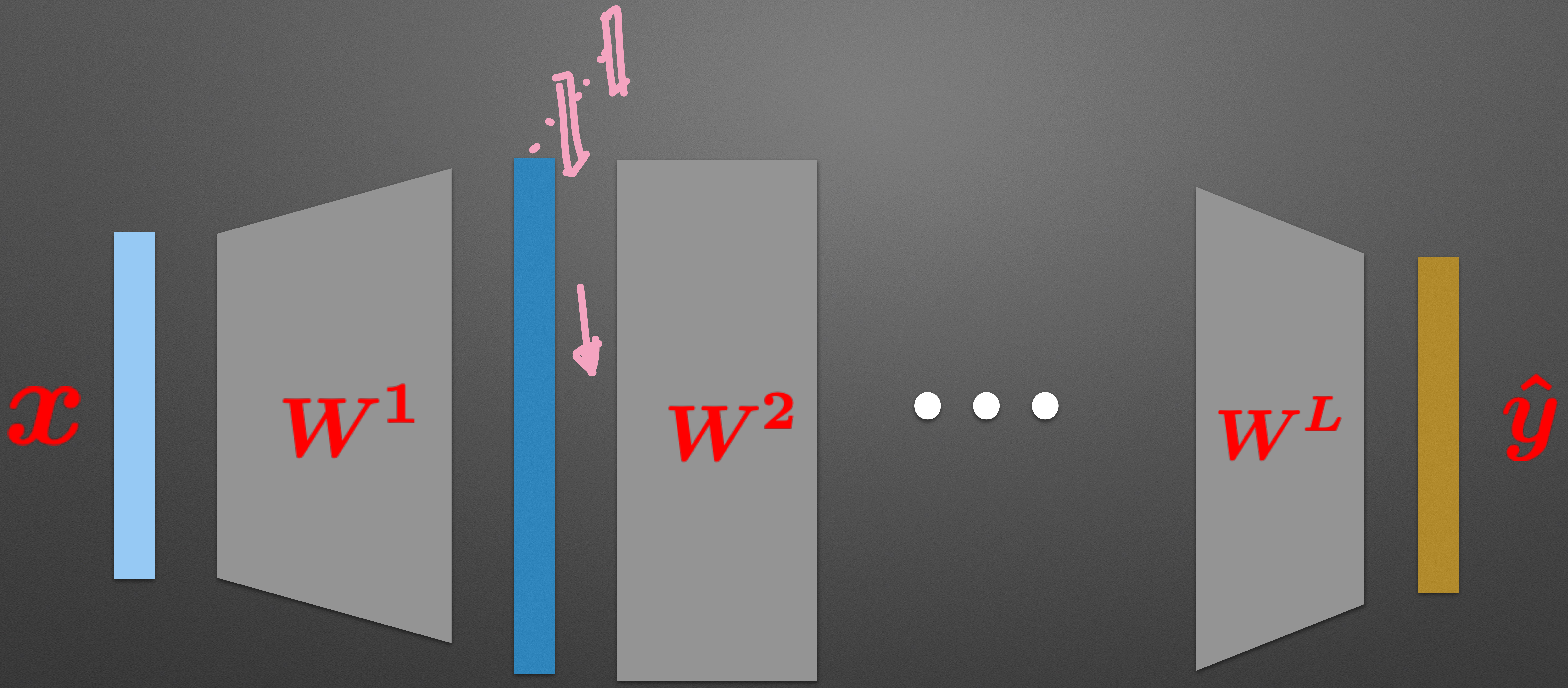
- Training data, validation data, test data.

Random



- Model training data - used for parameter learning.
- Validation data - used for hyper-parameter tuning (cross validation CV).

Feedforward networks



- Dense connections between the input and output - also called fully connected network.

Learning in feedforward networks

$\{x_1 \dots x_N\}$ $\{y_1 \dots y_N\}$

- Stochastic gradient descent (SGD) - Initialize the model parameters $\Theta^{0,0}$ (randomly)

iterations

mini-batch

for $q = \{1 \ldots Q\}$:

  for $b = \{1 \ldots B\}$:

$$\Theta^{q,b} = \Theta^{q,b-1} - \eta \frac{\partial E(\{\boldsymbol{y}_{b,1}, \ldots \boldsymbol{y}_{b,N_b}\}, \{\hat{\boldsymbol{y}}_{b,1}, \ldots \hat{\boldsymbol{y}}_{b,N_b}\})}{\partial \Theta}\Bigg|_{\Theta = \Theta^{q,b-1}}$$

$$\Theta^{q+1,0} = \Theta^{q,B}$$

return $\Theta^* = \Theta^{Q,B}$

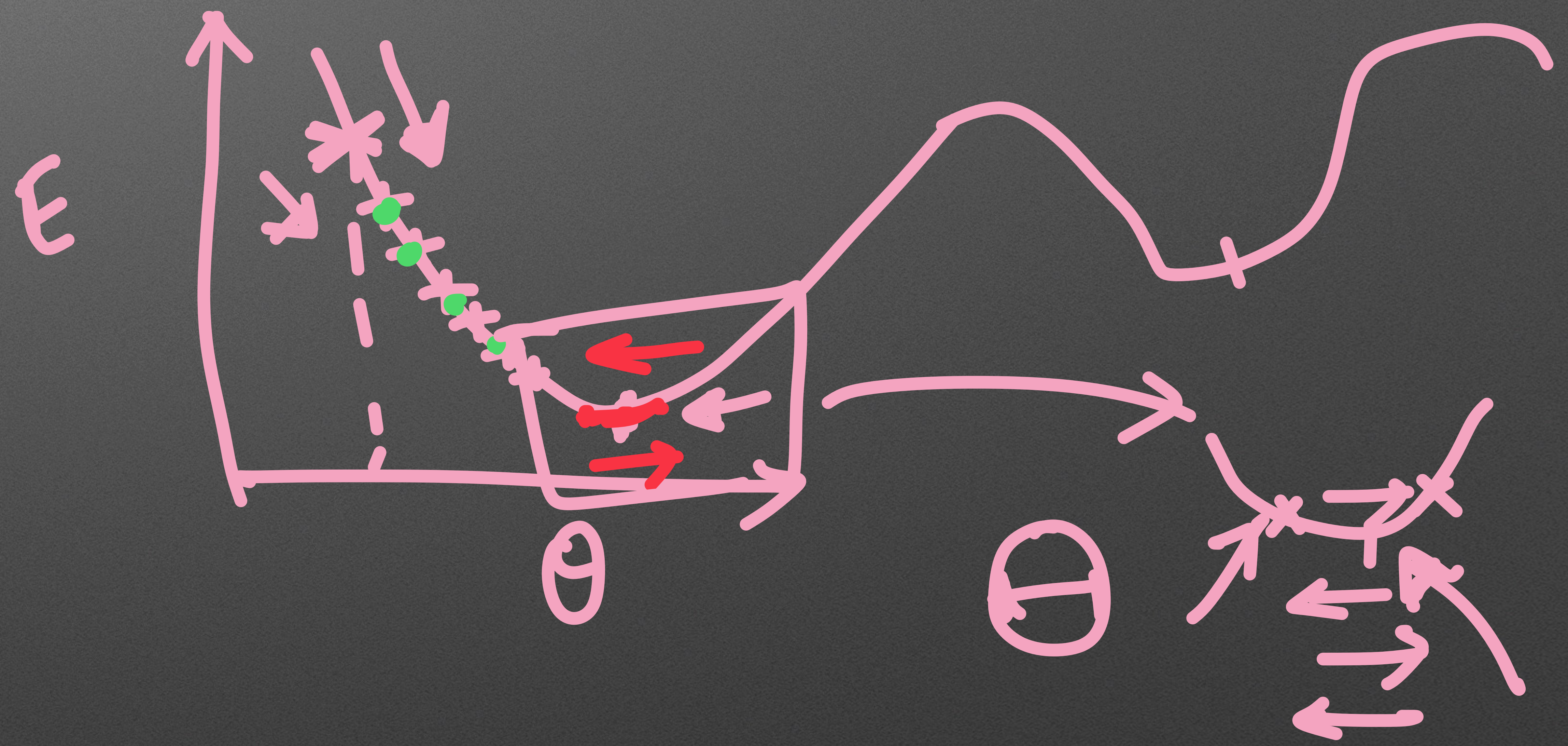
Convergence

Batch size	Speed	Grad. Acc.
↑	↓	↑
↓	↑	↓

Learning in feedforward networks

- Learning with momentum - accelerate the learning by adding a component of the previous gradient computation.

- RMSprop $\eta' = \dfrac{\eta}{RMS(g)}$



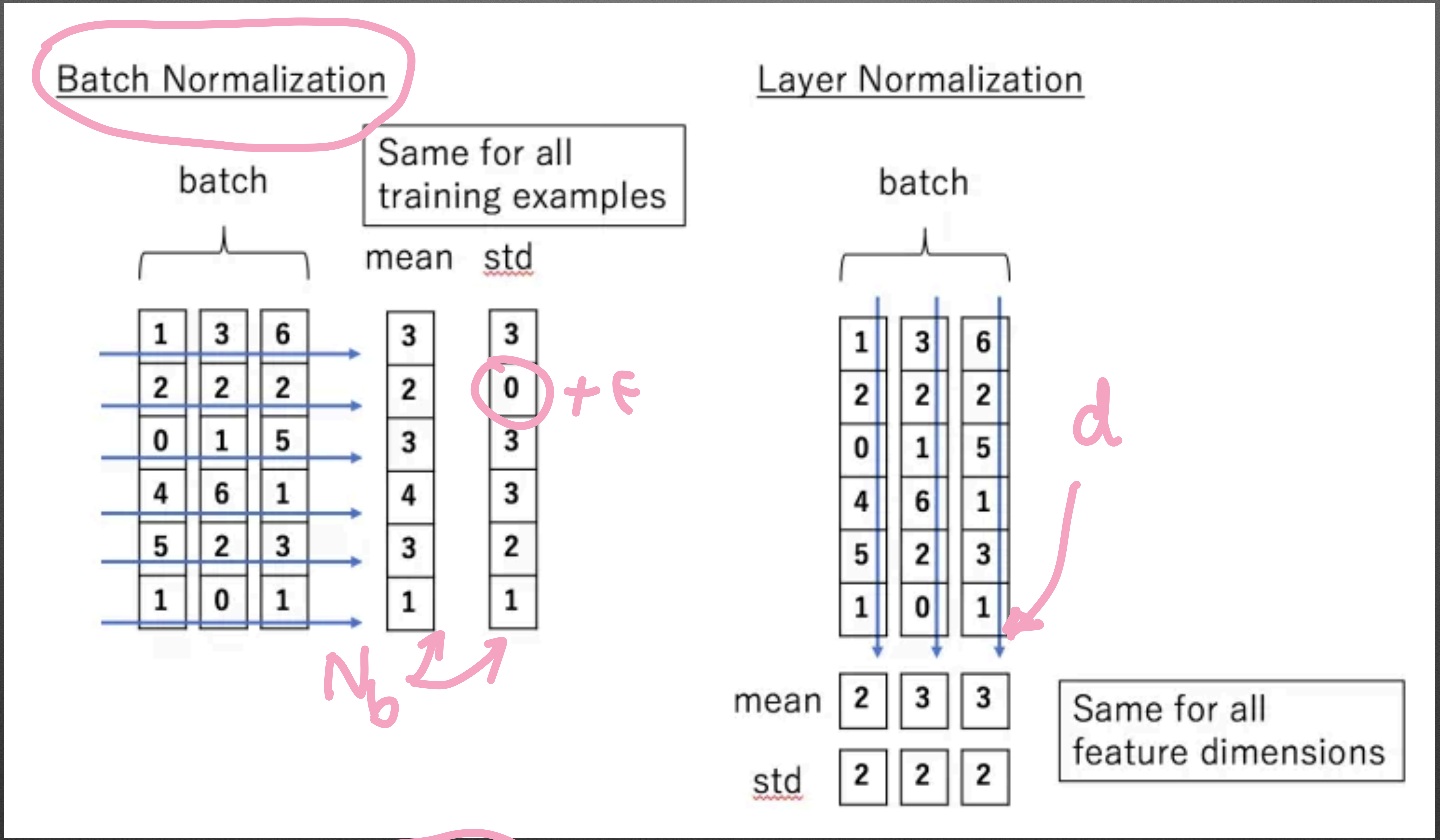
- Adam - adaptive moment estimation

  - combines momentum and RMSprop.

  - empirically shown to be effective in many applications.

Reading assignment - Overview of gradient descent algorithms
https://arxiv.org/pdf/1609.04747.pdf

Normalization



Reading assignment - How does batchnorm help optimization

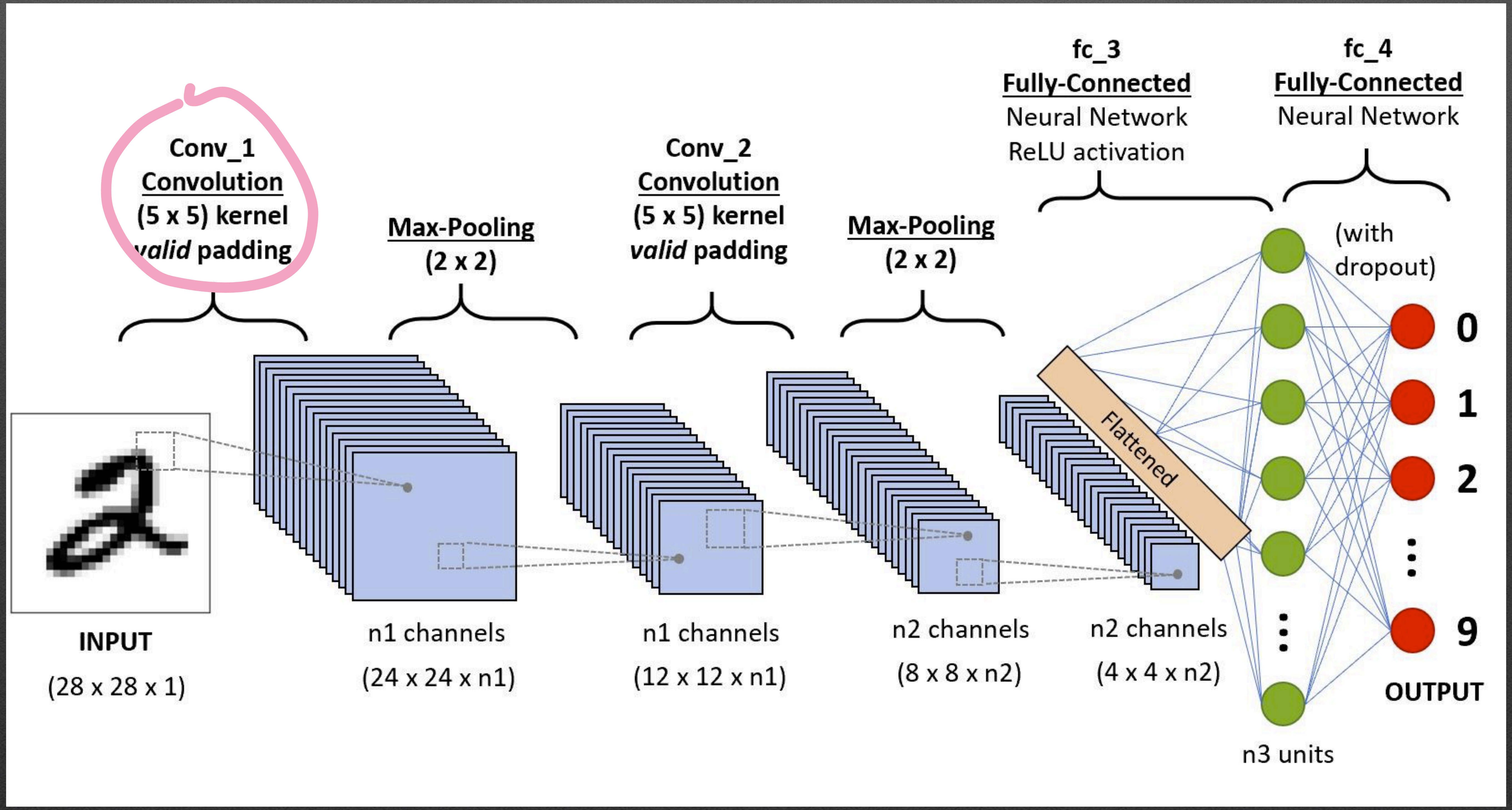
https://arxiv.org/pdf/1805.11604.pdf

Convolutional neural networks

- Replacing affine transformations with convolutional operations

$$H(m,n) = X * W(m,n) = \sum_{i,j=1,1}^{P,Q} X(m+i,n+j)W(i,j)$$

- Usually used with max-pooling based sub-sampling



Source - towardsdatascience.com

Module - I Visual and Time Series Modeling

Why do need recurrent models

- Learn from ordered pairs of $\boldsymbol{x}, \boldsymbol{y}$
  - ✓ All the data samples are treated independently.
    - ★ Data are shuffled before mini-batch formation
- If the input data and output labels are time-series data $\boldsymbol{x}(t), \boldsymbol{y}(t)$
  - ◉ DNNs/CNNs may fail to model the correlation of the data across the time
  - ◉ Question - how can we build models that capture the time evolution of the data and the labels.

Why do need recurrent models

- An interesting subset of this problem is where the input alone is a time series $\boldsymbol{x}(t), \boldsymbol{y}$ or have different indices $\boldsymbol{x}(t), \boldsymbol{y}(u)$

- Examples

  ✓ Text sequences

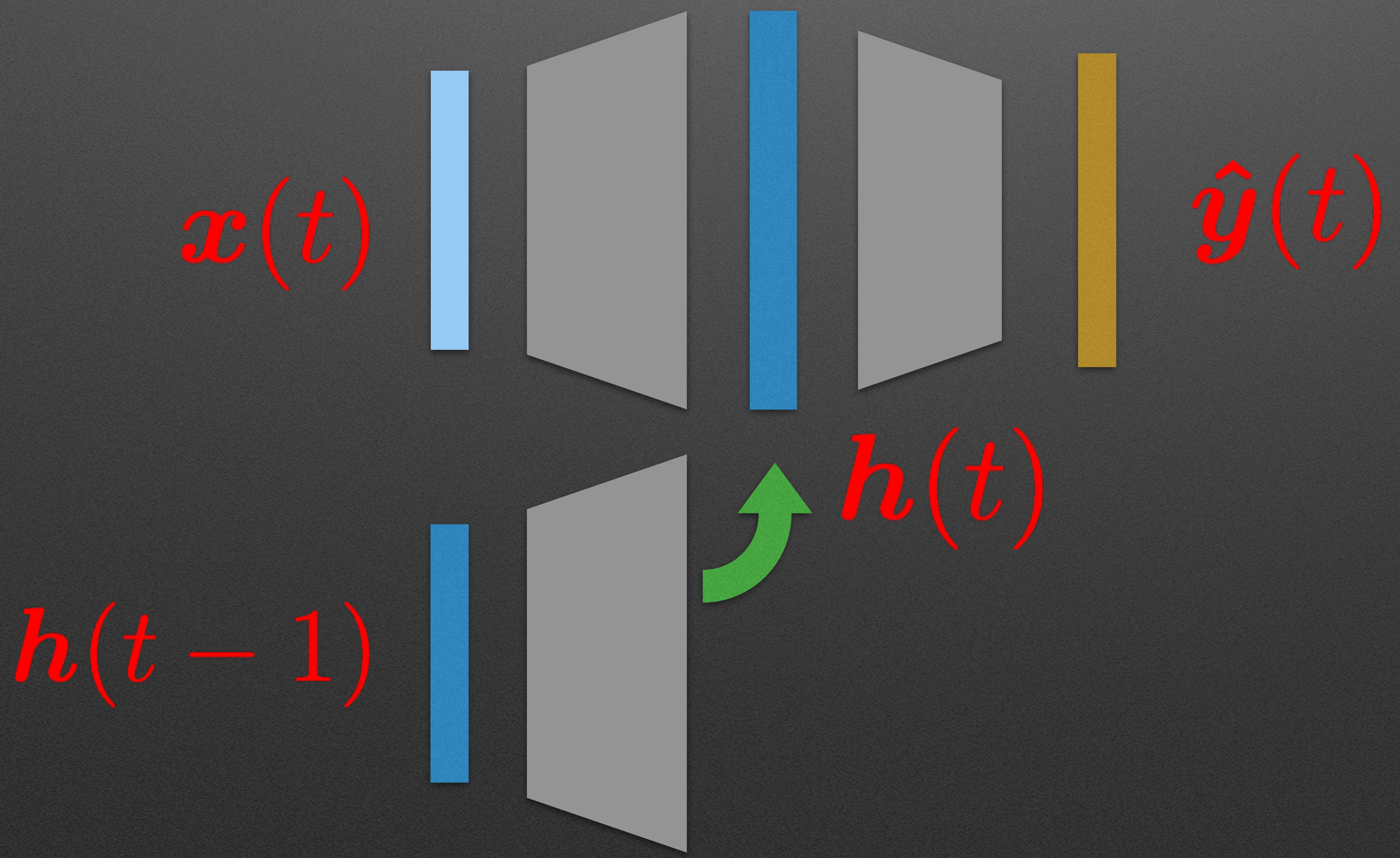
  ✓ Speech and audio

  ✓ Video sequences

First order recurrence - hidden layer

- Making the hidden layer a function of the previous outputs from the hidden layer along with the input

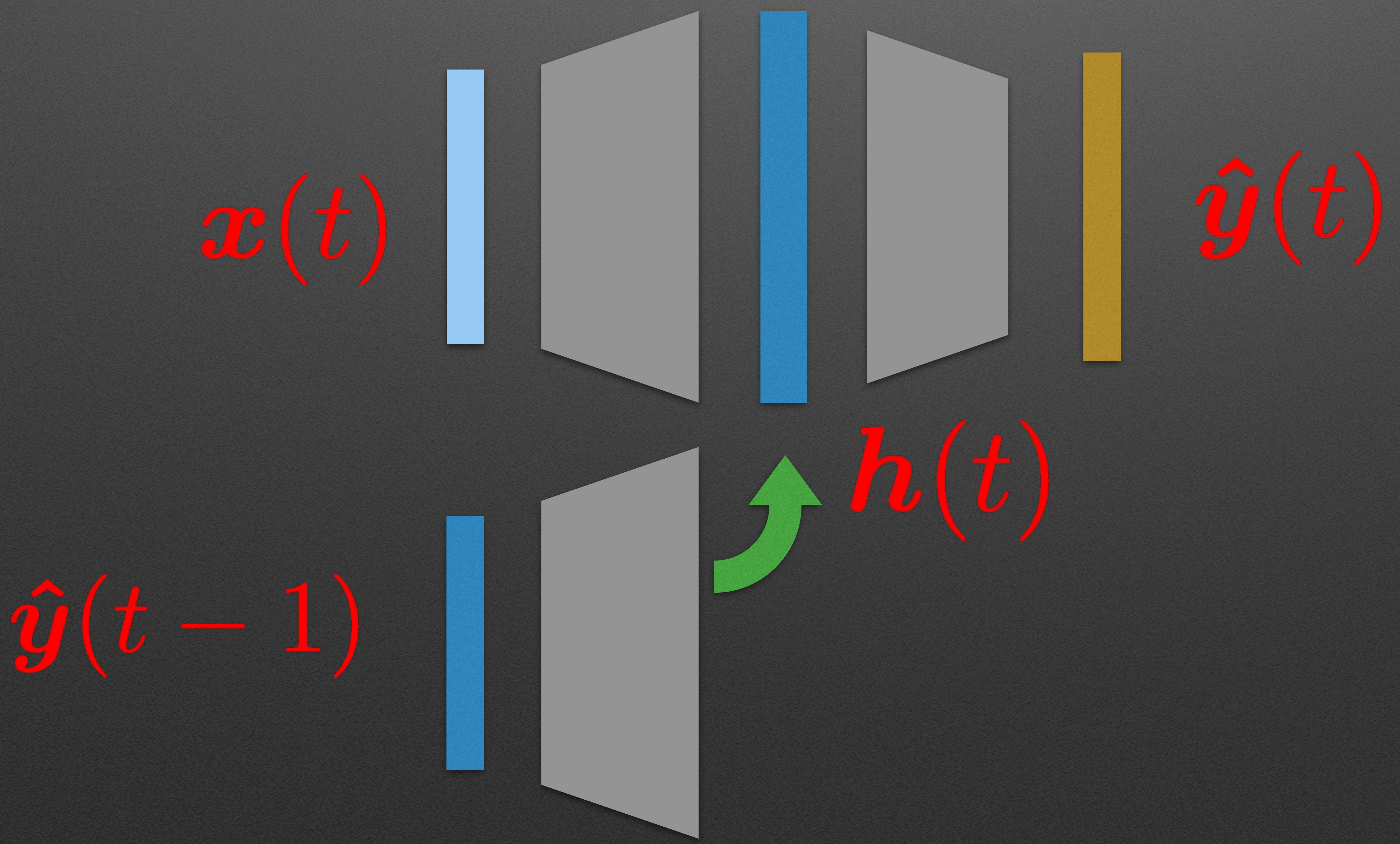
$$\boldsymbol{h}(t) = f\left(\boldsymbol{h}(t-1), \boldsymbol{x}(t)\right)$$



First order recurrence - output layer

- Making the hidden layer a function of the previous outputs from the hidden layer along with the input

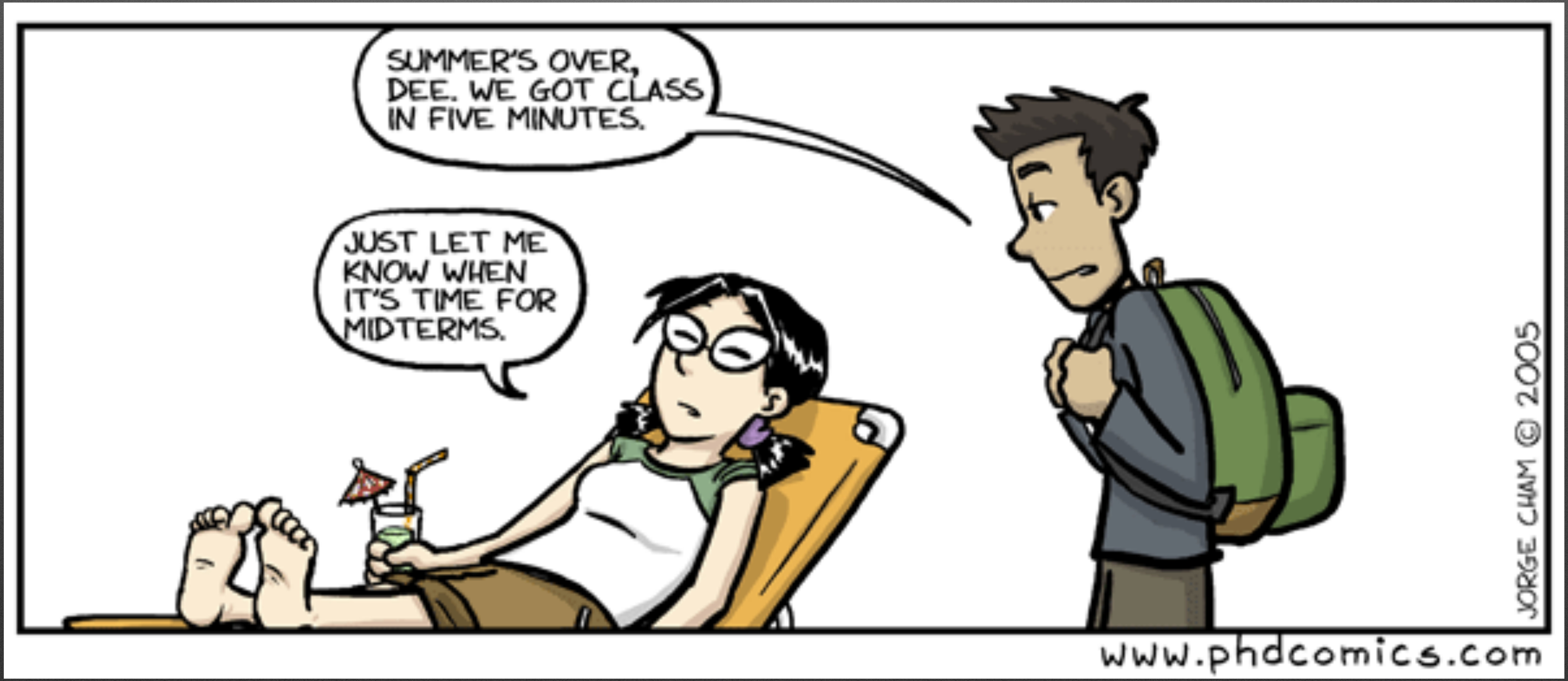
$$\boldsymbol{h}(t) = f\left(\hat{\boldsymbol{y}}(t-1), \boldsymbol{x}(t)\right)$$



Looking forward (next lecture)

- Learning in recurrence networks: Back-propagation in time.
- Unsupervised Learning: Issues with forgetting and long-short-term memory networks

The bell has rung



http://phdcomics.com/