

# Lecture 5: Learning Sequential Patterns with RNN & LSTM

Assoc Prof Tham Chen Khong

Dept of Electrical & Computer Engineering (ECE)

NUS

E-mail: [eletck@nus.edu.sg](mailto:eletck@nus.edu.sg)

*Acknowledgement:* some materials from Géron, Hands On ML



EEEC4400 Data Engineering and Deep Learning  
CK Tham, ECE NUS

## Overview

- Recurrent Neural Network (RNN)
- Long Short Term Memory (LSTM)
- Gated Recurrent Unit (GRU)

# Introduction

- Here, we consider neural networks that can process
  - sequences / sequential data
  - time series
- and predict the next outcomes
- Examples: speech, stock market, trajectory of moving objects, language translation (e.g. Google Translate)

## Recurrent Neurons and Layers

- Input and target (or output) values are presented to the network at each time step
- There are internal connections from one time step to the next time step

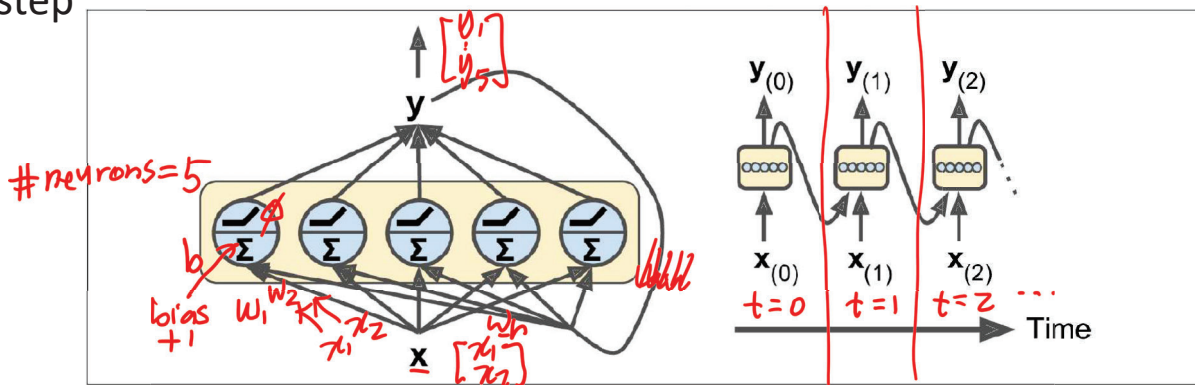


Figure 15-2. A layer of recurrent neurons (left) unrolled through time (right)

# RNN

Each recurrent neuron has two sets of weights: one for the inputs  $\mathbf{x}_{(t)}$  and the other for the outputs of the previous time step,  $\mathbf{y}_{(t-1)}$ . Let's call these weight vectors  $\mathbf{w}_x$  and  $\mathbf{w}_y$ . If we consider the whole recurrent layer instead of just one recurrent neuron, we can place all the weight vectors in two weight matrices,  $\mathbf{W}_x$  and  $\mathbf{W}_y$ . The output vector of the whole recurrent layer can then be computed pretty much as you might expect, as shown in Equation 15-1 ( $\mathbf{b}$  is the bias vector and  $\phi(\cdot)$  is the activation function (e.g., ReLU<sup>1</sup>).

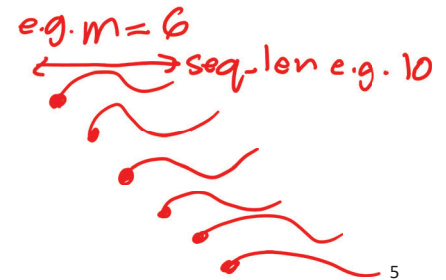
Equation 15-1. Output of a recurrent layer for a single instance

$$\mathbf{y}_{(t)} = \phi(\mathbf{W}_x^T \mathbf{x}_{(t)} + \mathbf{W}_y^T \mathbf{y}_{(t-1)} + \mathbf{b})$$

Equation 15-2. Outputs of a layer of recurrent neurons for all instances in a mini-batch

$$\mathbf{Y}_{(t)} = \phi(\mathbf{X}_{(t)} \mathbf{W}_x + \mathbf{Y}_{(t-1)} \mathbf{W}_y + \mathbf{b})$$

$$= \phi\left(\begin{bmatrix} \mathbf{X}_{(t)} & \mathbf{Y}_{(t-1)} \end{bmatrix} \mathbf{W} + \mathbf{b}\right) \text{ with } \mathbf{W} = \begin{bmatrix} \mathbf{W}_x \\ \mathbf{W}_y \end{bmatrix}$$



In this equation:

# RNN

- $\mathbf{Y}_{(t)}$  is an  $m \times n_{\text{neurons}}$  matrix containing the layer's outputs at time step  $t$  for each instance in the mini-batch ( $m$  is the number of instances in the mini-batch and  $n_{\text{neurons}}$  is the number of neurons).
- $\mathbf{X}_{(t)}$  is an  $m \times n_{\text{inputs}}$  matrix containing the inputs for all instances ( $n_{\text{inputs}}$  is the number of input features).
- $\mathbf{W}_x$  is an  $n_{\text{inputs}} \times n_{\text{neurons}}$  matrix containing the connection weights for the inputs of the current time step.
- $\mathbf{W}_y$  is an  $n_{\text{neurons}} \times n_{\text{neurons}}$  matrix containing the connection weights for the outputs of the previous time step.
- $\mathbf{b}$  is a vector of size  $n_{\text{neurons}}$  containing each neuron's bias term.
- The weight matrices  $\mathbf{W}_x$  and  $\mathbf{W}_y$  are often concatenated vertically into a single weight matrix  $\mathbf{W}$  of shape  $(n_{\text{inputs}} + n_{\text{neurons}}) \times n_{\text{neurons}}$  (see the second line of Equation 15-2).
- The notation  $[\mathbf{X}_{(t)} \mathbf{Y}_{(t-1)}]$  represents the horizontal concatenation of the matrices  $\mathbf{X}_{(t)}$  and  $\mathbf{Y}_{(t-1)}$ .

Notice that  $\mathbf{Y}_{(t)}$  is a function of  $\mathbf{X}_{(t)}$  and  $\mathbf{Y}_{(t-1)}$ , which is a function of  $\mathbf{X}_{(t-1)}$  and  $\mathbf{Y}_{(t-2)}$ , which is a function of  $\mathbf{X}_{(t-2)}$  and  $\mathbf{Y}_{(t-3)}$ , and so on. This makes  $\mathbf{Y}_{(t)}$  a function of all the inputs since time  $t = 0$  (that is,  $\mathbf{X}_{(0)}$ ,  $\mathbf{X}_{(1)}$ , ...,  $\mathbf{X}_{(t)}$ ). At the first time step,  $t = 0$ , there are no previous outputs, so they are typically assumed to be all zeros.