Edge Artificial Intelligence Design

by

Gawsalyan Sivapalan

School of Electrical & Electronic Engineering
University College Dublin
Belfield, Dublin 4
Ireland

April 2021

Contents

Contents i

1 ANN: *Artificial Neural Net* **1**
1.0.1 Fixed Point Implementation 1

2 Machine Learning Toolbox Design **5**
2.1 Standard Floating Point Implementation 7
2.1.1 MLP Node 7
2.1.2 LSTM Cell 9
2.2 Fixed Point Implementation 16
2.2.1 MLP Node 16
2.2.2 LSTM Cell 18

Chapter 1

ANN

Artificial Neural Net

1.0.1 Fixed Point Implementation

The neural network training is meticulously designed using a signed fixed-point architecture with 12 fractional bits, and later constraint to 6 fractional bits, compatible with the targeted embedded AI system to allow space for calculation of large matrix.

Qm.n Number Format

Q is one of several widely used binary fixed-point number formats where the number of fractional bits (n) and the number of integer bits (m) is specified. Signed Q values are stored in two's complement format, just like signed integer values on most processors. In two's complement, the sign bit is extended to the register size.

For a given Qm.n format, using an m+n bit signed integer container with n fractional bits:

- its range is $-(2^{m-1}), 2^{m-1} - 2^{-n}$
- its resolution is 2^{-n}

Activation Functions

In pursuance to train and predict with a fixed-point neural network, the relevant activation and gate functions of the cells should be changed accordingly to accommodate fixed-point calculations. These functions are usually constrained by a pair of horizontal asymptotes and define the output of artificial neurons. The fast approximate version of these functions and its derivative used in the back-propagation training are discussed next.

Sigmoid Activation Function (σ)

It is one of the essential cell activation functions used across many popular neural networks. The below equations shows the original sigmoid and its derivative used to tune the weights of the neurons in an ordinary network.

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

$$\partial\sigma(x) = \sigma(x) * (1 - \sigma(x))$$

The above sigmoidal function cannot be directly used with fixed-point values as exponential functions are designed not to work with fixed-point architecture. Besides, the absolute exponential calculation is very costly and it is almost impossible to deploy in an embedded system without approximation. Therefore, it is translated to an approximate fast version of the function to facilitate the operation in an embedded environment with fixed-point architecture as given below.

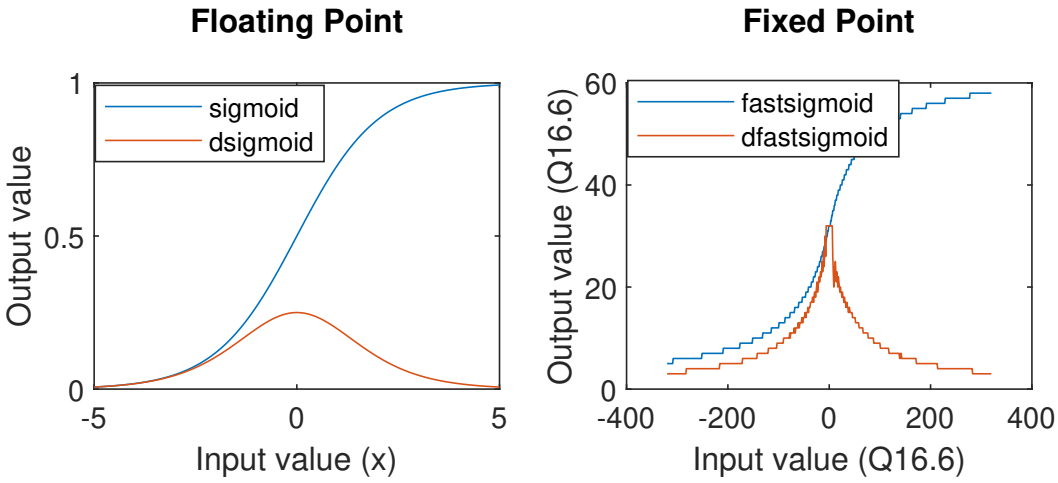


Figure 1.1: Floating and Fixed point approximation of Sigmoid function

Fixed Point (Q26.6) implementation of Sigmoid:

$$\Sigma(x) = \frac{x << 6}{64 + |x|}$$

and its derivative:

x > 0: $\partial\Sigma(x) = \frac{1}{x} * (\Sigma(x) - 32) * (64 - (\Sigma(x) - 32))$

x $= 0$: $\partial\Sigma(x) = 32$

x < 0: $\partial\Sigma(x) = \frac{1}{x} * (\Sigma(x) - 32) * (64 + (\Sigma(x) - 32))$

'$<<$' and '$>>$' represents bit shift to left and right respectively.
The value 64 represents integer value of 1 in Q26.6 format.
∂ represents the derivative function.

Tanh Activation Function

It is mainly used inside the LSTM cells as candidate gate Ct (Figure **??**) activation function. The original equation and its derivative are provided below along with its fast approximate fixed point implementation used in this work.

$$\tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1}$$

$$\partial tanh(x) = 1 - tanh(x)^2$$

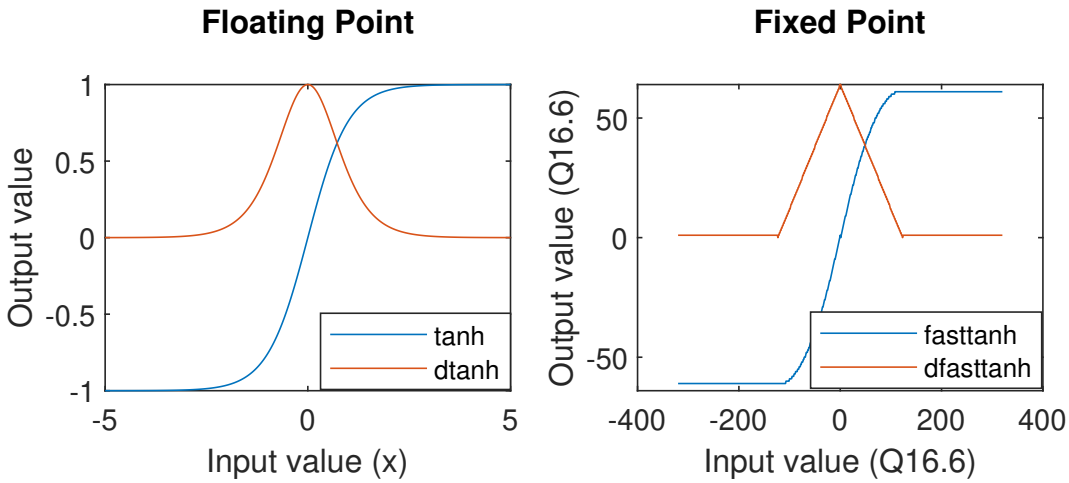


Figure 1.2: Floating and Fixed point approximation of Tanh function

Fixed Point (Q26.6) implementation of Tanh:

x $>$ 123:

$$fast\tanh(x) = 61$$

$$\partial fast\tanh(x) = 0$$

123 $>$ x $>$ 0:

$$fast\tanh(x) = 61 - (17 * ((x - 123)^2 >> 6) >> 6)$$

$$\partial fast\tanh(x) = -2 * (17 * (x - 123) >> 6)$$

0 $>$ x $>$ -123:

$$fast\tanh(x) = (17 * ((x + 123)^2 >> 6) >> 6) - 61$$

$$\partial fast\tanh(x) = 2 * (17 * (x + 123) >> 6)$$

-123 $>$ x:

$$fast\tanh(x) = -61$$

$$\partial fast\tanh(x) = 0$$

'$<<$' and '$>>$' represents bit shift to left and right respectively.
The value 64 represents integer value of 1 in Q26.6 format.
∂ represents the derivative function.

Softmax function

This is primarily used in the classification layers. It is not related to single-neuron rather the calculation is performed over every class node in the particular layer.

$$softMAX(x_i) = \frac{e^(x_i)}{\sum e^(x_i)}$$

$$\partial softMAX(x_i) = softMAX(x_i) * (1 - softMAX(x_i))$$

Fixed Point (Q26.6) implementation of SoftMAX:

$$fastsoftMAX(x) = \frac{(64 + x_i) << 6}{\sum(64 + x_i)}$$

$$\begin{aligned} \partial fastsoftMAX(x) &= \frac{64}{64 + x_i} * fastsoftMAX(x_i) \\ &\quad *(1 - fastsoftMAX(x_i)) \end{aligned}$$

'$<<$' and '$>>$' represents bit shift to left and right respectively.
The value 64 represents integer value of 1 in Q26.6 format.
∂ represents the derivative function.

Chapter 2

Machine Learning Toolbox Design

The Chapter describes the mathematical equation for the forward and backward propagation in the Artificial Neural Network.

- Chapter 2.1 delineates regular equations with floating point implementation found uniformly across standard machine learning libraries.
  - 2.1.1: Multi Layer Perceptron (MLP) node forward predictions using the input X and backward error propagation for weight corrections ∂W and ∂b at the node.
  - 2.1.2: Long Short Term Memory (LSTM) cell forward prediction at each time step using the input X_{t-1} and backward error propagation for weight corrections ∂W_i, ∂b_i, ∂W_f, ∂b_f, ∂W_c, ∂b_c, ∂W_o, ∂b_o, ∂W_y, and ∂b_y at the LSTM cell.
- Chapter 2.2 delineates proposed equations with fixed-point implementation used in our research. Fast approximated functions and their derivatives are discussed in Chapter 1.0.1.
  - 2.2.1: Multi Layer Perceptron (MLP) node forward predictions using the input X and backward error propagation for weight corrections ∂W and ∂b at the node level.
  - 2.2.2: Long Short Term Memory (LSTM) cell forward prediction at each time step using the input X_{t-1} and backward error propagation for weight corrections ∂W_i, ∂b_i, ∂W_f, ∂b_f, ∂W_c, ∂b_c, ∂W_o, ∂b_o, ∂W_y, and ∂b_y at the cell level.

The equations uses standard symbols as explained below and new updated Weights (W^*) and biases (b^*) are as follows,

$$W_x^* = W_x + \alpha * \partial W_x$$

$$b_x^* = b + \alpha * \partial b_x$$

where α is the learning rate of the network and $x \in \{i, f, c, o, y\}$ for LSTM and $x \in \{\emptyset\}$ for MLP.

- $\times$: Cross Product
- $\cdot$: Dot Product
- ∂ : Partial Differentiation
- I : Vector of all ones

- $'$: Transpose of the Matrix
- $X_{[a \times b]}$: Size of the vector a, b
- $f(x)$: function f of x

2.1 Standard Floating Point Implementation

2.1.1 MLP Node

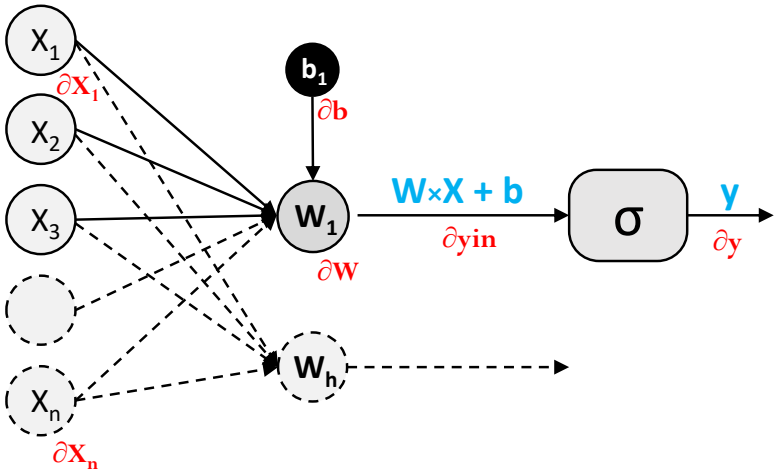


Figure 2.1: Forward and Back Propagation at MLP Node

n : Size of Input vector to MLP Layer
h : Number of Hidden nodes in the MLP Layer
$\sigma()/sigmoid()$: Sigmoid activation function (1.0.1)

Forward Prediction

$$y_{in_{[h\times 1]}} = W_{[h\times n]} \times X_{[n\times 1]} + b_{[h\times 1]} \tag{1.1.1}$$

$$\begin{aligned} y_{[h\times 1]} &= \sigma(y_{in_{[h\times 1]}}) \\ &= \sigma(W_{[h\times n]} \times X_{[n\times 1]} + b_{[h\times 1]}) \end{aligned} \tag{1.1.2}$$

Back Propagation

$$y = sigmoid(y_{in}) \tag{1.2.1}$$

Back Propagation error correction at MLP node

Partial differentiation of Eq.1.2.1 w.r.t y_{in}

$$\frac{\partial y}{\partial y_{in}} = y \cdot (I - y)$$

$$\partial y_{in} = \partial y \cdot y \cdot (I - y) \tag{1.2.2}$$

$$y_{in} = W \times X + b \tag{1.2.3}$$

Partial differentiation of Eq.1.2.3 w.r.t W

$$\frac{\partial y_{in}}{\partial W} = X$$

$$\begin{aligned} \partial W &= \partial y_{in} \times X' \\ &= \partial y \cdot y \cdot (I - y) \times X' \end{aligned} \tag{1.2.4}$$

Partial differentiation of Eq.1.2.3 w.r.t b

$$\frac{\partial y_{in}}{\partial b} = 1$$

$$\begin{aligned} \partial b &= \partial y_{in} \\ &= \partial y \cdot y \cdot (I - y) \end{aligned} \tag{1.2.5}$$

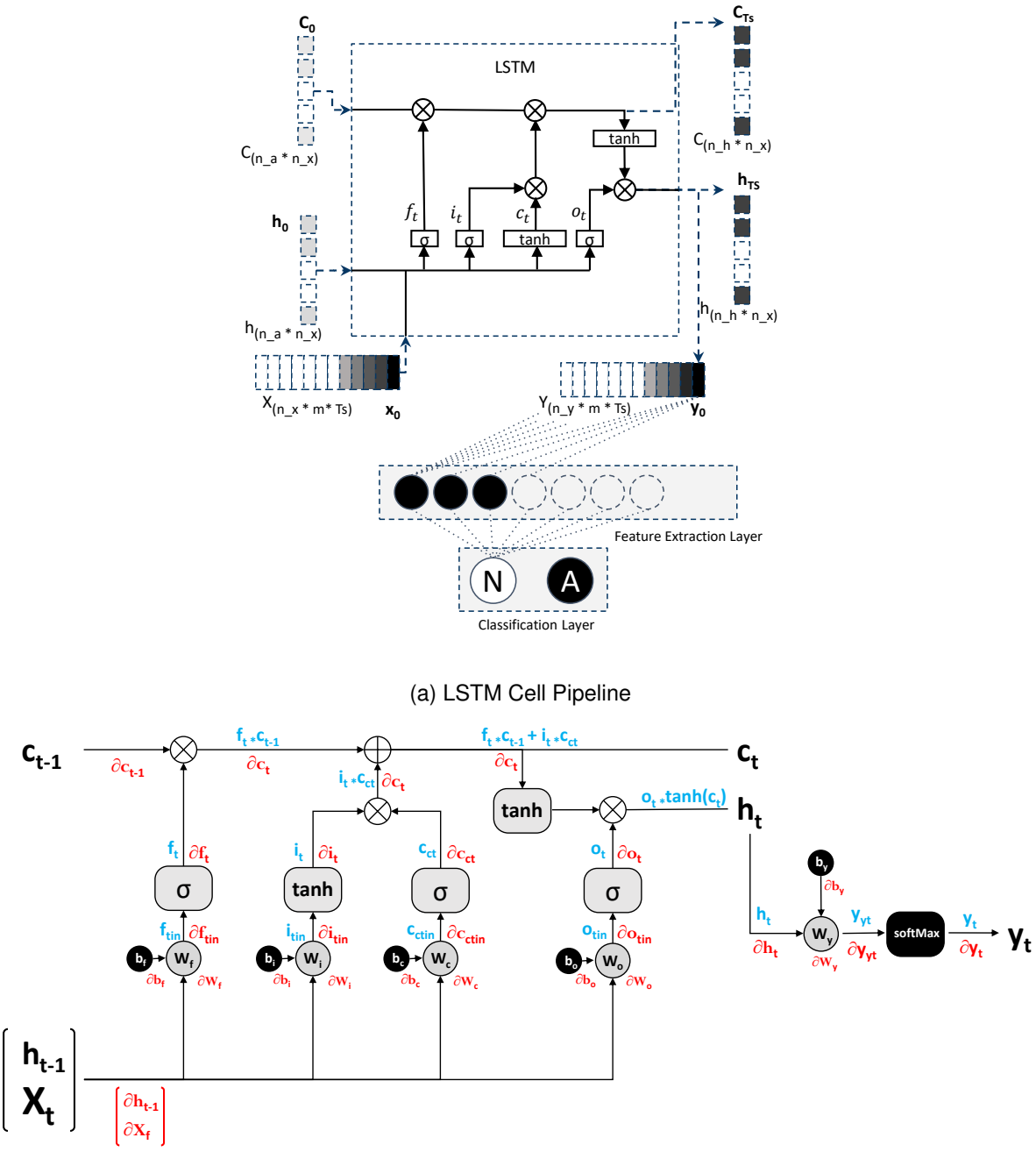
Back Propagation error to the previous layer

Partial differentiation of Eq.1.2.3 w.r.t X

$$\frac{\partial y_{in}}{\partial X} = W$$

$$\begin{aligned} \partial X &= W' \times \partial y_{in} \\ &= W' \times (\partial y \cdot y \cdot (I - y)) \end{aligned} \tag{1.2.6}$$

2.1.2 LSTM Cell



(a) LSTM Cell Pipeline

(b) Forward and Back Propagation through single time step at LSTM Cell

Figure 2.2: Long Short Term Memory

n_h	: Size of Hidden State vector of single LSTM cell
n_f	: Size of One hot vector representation of element in input vector X (default n_f = 1)
n_x	: Size of Input vector X
$\sigma()/sigmoid()$	: Sigmoid activation function (1.0.1)
$\tanh()$	: Tanh activation function (1.0.1)
$softMax()$	: SoftMax function (1.0.1)

Forward Prediction

Forget gate f_t:

$$f_{t_{[n_h \times n_f]}} = \sigma(W_{f_{[n_h \times (n_h+n_x)]}} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}_{[(n_h+n_x) \times n_f]} + b_{f_{[n_h \times n_f]}}) \tag{2.1.1}$$

Input gate i_t:

$$i_{t_{[n_h \times n_f]}} = \sigma(W_{i_{[n_h \times (n_h+n_x)]}} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}_{[(n_h+n_x) \times n_f]} + b_{i_{[n_h \times n_f]}}) \tag{2.1.2}$$

Candidate gate c_{ct}:

$$c_{ct_{[n_h \times n_f]}} = \sigma(W_{c_{[n_h \times (n_h+n_x)]}} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}_{[(n_h+n_x) \times n_f]} + b_{c_{[n_h \times n_f]}}) \tag{2.1.3}$$

Cell state c_t:

$$c_{t_{[n_h \times n_f]}} = f_{t_{[n_h \times n_f]}} \times c_{t-1_{[n_h \times n_f]}} + i_{t_{[n_h \times n_f]}} \times c_{ct_{[n_h \times n_f]}} \tag{2.1.4}$$

Output gate o_t:

$$o_{t_{[n_h \times n_f]}} = \sigma(W_{o_{[n_h \times (n_h+n_x)]}} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}_{[(n_h+n_x) \times n_f]} + b_{o_{[n_h \times n_f]}}) \tag{2.1.5}$$

Hidden state h_t:

$$h_{t_{[n_h \times n_f]}} = o_{t_{[n_h \times n_f]}} \times \tanh(c_t)_{[n_h \times n_f]} \tag{2.1.6}$$

Output y_t:

$$y_{yt_{[n_y \times 1]}} = W_{y_{[n_y \times n_h]}} \times h_{t_{[n_h \times 1]}} + b_{y_{[n_y \times 1]}} \tag{2.1.7}$$

$$y_{t_{[n_y \times 1]}} = SoftMax(y_{yt_{[n_y \times 1]}}) \tag{2.1.8}$$

Back Propagation

$$y_t = softMax(y_{yt}) \tag{2.2.1}$$

Back Propagation error correction at Output (y)

Partial differentiation of Eq.2.2.1 w.r.t y_{yt}

$$\frac{\partial y_t}{\partial y_{yt}} = y_t \cdot (I - y_t)$$

$$\partial y_{yt} = \partial y_t \cdot y_t \cdot (I - y_t) \tag{2.2.2}$$

$$y_{yt} = W_y \times h_t + b_y \tag{2.2.3}$$

Partial differentiation of Eq.2.2.3 w.r.t W_y

$$\frac{\partial y_{yt}}{\partial W_y} = h_t$$

$$\begin{aligned} \partial W_y &= \partial y_{yt} \times h_t' \\ &= (\partial y_t \cdot y_t \cdot (I - y_t)) \times h_t' \end{aligned} \tag{2.2.4}$$

Partial differentiation of Eq.2.2.3 w.r.t b_y

$$\frac{\partial y_{yt}}{\partial b_y} = 1$$

$$\begin{aligned} \partial b_y &= \partial y_{yt} \\ &= \partial y_t \cdot y_t \cdot (I - y_t) \end{aligned} \tag{2.2.5}$$

Back Propagation error in Hidden State (h)

Partial differentiation of Eq.2.2.3 w.r.t h_t

$$\frac{\partial y_{yt}}{\partial h_t} = W_y$$

$$\partial h_t = W_y' \times \partial y_{yt} + \delta h_t$$

where $\delta h_t = 0$ for final time step (Ts) and ∂h_{t+1} for the rest

$$\partial h_t = W_y' \times (\partial y_t \cdot y_t \cdot (I - y_t)) + \delta h_t \tag{2.2.6}$$

$$h_t = o_t \cdot \tanh(c_t) \quad \text{(2.2.7)}$$
$$\text{where } o_t = sigmoid(o_{tin})$$
$$c_t = \tanh(c_{tin})$$

Back Propagation error correction at Output Gate (o)

Partial differentiation of Eq.2.2.7 w.r.t o_{tin}

$$\frac{\partial h_t}{\partial o_{t_{in}}} = o_t \cdot (I - o_t) \cdot \tanh(c_t)$$

$$\partial o_{t_{in}} = \partial h_t \cdot o_t \cdot (I - o_t) \cdot \tanh(c_t) \quad \text{(2.2.8)}$$

$$o_{tin} = W_o \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix} + b_o \quad \text{(2.2.9)}$$

Partial differentiation of Eq.2.2.9 w.r.t W_o

$$\frac{\partial o_{tin}}{\partial W_o} = \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}$$

$$\begin{aligned} \partial W_o &= \partial o_{tin} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}' \\ &= (\partial h_t \cdot o_t \cdot (I - o_t) \cdot tanh(c_t)) \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}' \\ &= (W_y' \times ((\partial y_t \cdot y_t \cdot (I - y_t))) \cdot o_t \cdot (I - o_t) \cdot tanh(c_t)) \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}' \end{aligned} \quad \text{(2.2.10)}$$

Partial differentiation of Eq.2.2.9 w.r.t b_o

$$\frac{\partial o_{tin}}{\partial b_o} = 1$$

$$\begin{aligned} \partial b_o &= \partial o_{tin} \\ &= \partial h_t \cdot o_t \cdot (I - o_t) \cdot \tanh(c_t) \\ &= W_y' \times ((\partial y_t \cdot y_t \cdot (I - y_t))) \cdot o_t \cdot (I - o_t) \cdot \tanh(c_t) \end{aligned} \quad \text{(2.2.11)}$$

Back Propagation error in Cell State (c)

Partial differentiation of Eq.2.2.7 w.r.t c_t

$$\frac{\partial h_t}{\partial c_t} = o_t \cdot (I - \tanh^2(c_t))$$

$$\partial c_t = \partial h_t \cdot o_t \cdot (I - \tanh^2(c_t)) + \delta c_t \tag{2.2.12}$$

where $\delta c_t = 0$ for final time step (Ts) and ∂c_{t+1} for the rest

$$c_t = f_t \cdot c_{t-1} + i_t \cdot c_{ct} \tag{2.2.13}$$

Back Propagation error correction at Candidate Gate (c_c)

Partial differentiation of Eq.2.2.13 w.r.t c_{ctin}

$$\begin{aligned} \frac{\partial c_t}{\partial c_{ctin}} &= i_t \cdot \frac{\partial \tanh(c_{ctin})}{\partial c_{ctin}} \\ &= i_t \cdot (I - \tanh^2(c_{ctin})) \\ &= i_t \cdot (I - c_{ct}^2) \end{aligned}$$

$$\partial c_{ctin} = \partial c_t \cdot i_t \cdot (I - c_{ct}^2) \tag{2.2.14}$$

$$c_{ctin} = W_c \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix} + b_c \tag{2.2.15}$$

Partial differentiation of Eq.2.2.15 w.r.t W_c

$$\begin{aligned} \partial W_c &= \partial c_{ctin} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}' \\ &= (\partial c_t \cdot i_t \cdot (I - c_{ct}^2)) \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}' \end{aligned} \tag{2.2.16}$$

Partial differentiation of Eq.2.2.15 w.r.t b_c

$$\begin{aligned} \partial b_c &= \partial c_{ctin} \\ &= \partial c_t \cdot i_t \cdot (I - c_{ct}^2) \end{aligned} \tag{2.2.17}$$

Back Propagation error correction at Input Gate (i)

Partial differentiation of Eq.2.2.13 w.r.t i_{tin}

$$\begin{aligned}\frac{\partial c_t}{\partial i_{tin}} &= \frac{\partial sigmoid(i_{tin})}{\partial i_{tin}} \cdot c_{ct} \\ &= sigmoid(i_{tin}) \cdot (I - sigmoid(i_{tin})) \cdot c_{ct} \\ &= i_t \cdot (I - i_t) \cdot c_{ct}\end{aligned}$$

$$\partial i_{tin} = \partial c_t \cdot i_t \cdot (I - i_t) \cdot c_{ct} \tag{2.2.18}$$

$$i_{tin} = W_i \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix} + b_i \tag{2.2.19}$$

Partial differentiation of Eq.2.2.19 w.r.t W_i

$$\frac{\partial i_{tin}}{\partial W_i} = \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}$$

$$\begin{aligned}\partial W_i &= \partial i_{tin} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}' \\ &= \partial c_t \cdot i_t \cdot (I - i_t) \cdot c_{ct} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}'\end{aligned} \tag{2.2.20}$$

Partial differentiation of Eq.2.2.19 w.r.t b_i

$$\frac{\partial i_{tin}}{\partial b_i} = 1$$

$$\begin{aligned}\partial b_i &= \partial i_{tin} \\ &= \partial c_t \cdot i_t \cdot (I - i_t) \cdot c_{ct}\end{aligned} \tag{2.2.21}$$

Back Propagation error correction at Forget Gate (f)

Partial differentiation of Eq.2.2.13 w.r.t f_{tin}

$$\begin{aligned}\frac{\partial c_t}{\partial f_{tin}} &= \frac{\partial sigmoid(f_{tin})}{\partial f_{tin}} \cdot c_{t-1} \\ &= sigmoid(f_{tin}) \cdot (I - sigmoid(f_{tin})) \cdot c_{t-1} \\ &= f_t \cdot (I - f_t) \cdot c_{t-1}\end{aligned}$$

$$\partial f_{tin} = \partial c_t \cdot f_t \cdot (I - f_t) \cdot c_{t-1} \tag{2.2.22}$$

$$f_{tin} = W_f \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix} + b_f \tag{2.2.23}$$

Partial differentiation of Eq.2.2.23 w.r.t W_f

$$\frac{\partial f_{tin}}{\partial W_f} = \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}$$

$$\begin{aligned} \partial W_f &= \partial f_{tin} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}' \\ &= \partial c_t \cdot f_t \cdot (I - f_t) \cdot c_{t-1} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}' \end{aligned} \tag{2.2.24}$$

Partial differentiation of Eq.2.2.23 w.r.t b_f

$$\frac{\partial f_{tin}}{\partial b_f} = 1$$

$$\begin{aligned} \partial b_f &= \partial f_{tin} \\ &= \partial c_t \cdot f_t \cdot (I - f_t) \cdot c_{t-1} \end{aligned} \tag{2.2.25}$$

Back Propagation error correction at h_{t-1}

$$\frac{\partial \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}}{\partial W} = \partial f_{tin} + \partial i_{tin} + \partial c_{ctin} + \partial o_{tin} \tag{2.2.26}$$

$$\partial \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix} = W_f' \times \partial f_{tin} + W_i' \times \partial i_{tin} + W_c' \times \partial c_{ctin} + W_o' \times \partial o_{tin} \tag{2.2.27}$$

Back Propagation error correction at c_{t-1}

$$\frac{\partial c_t}{\partial c_{t-1}} = f_t \tag{2.2.28}$$

$$\partial c_{t-1} = \partial c_t \cdot f_t \tag{2.2.29}$$

2.2 Fixed Point Implementation

2.2.1 MLP Node

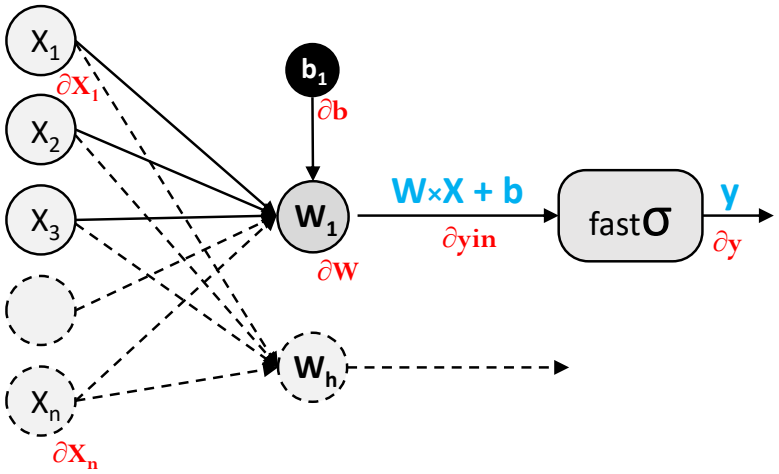


Figure 2.3: Forward and Back Propagation at Fixed Point MLP Node

n	: Size of Input vector to MLP Layer
h	: Number of Hidden nodes in the MLP Layer
$fast\sigma()/fastsigmoid()$	: fast approximation of Sigmoid activation function (1.0.1)
$\partial fast\sigma()/\partial fastsigmoid()$	: derivative of fast approximated Sigmoid activation function (1.0.1)

Forward Prediction

$$y_{in_{[h\times 1]}} = W_{[h\times n]} \times X_{[n\times 1]} + b_{[h\times 1]} \tag{3.1.1}$$

$$\begin{aligned} y_{[h\times 1]} &= fast\sigma(y_{in_{[h\times 1]}}) \\ &= fast\sigma(W_{[h\times n]} \times X_{[n\times 1]} + b_{[h\times 1]}) \end{aligned} \tag{3.1.2}$$

Back Propagation

$$y = fastsigmoid(y_{in}) \tag{3.2.1}$$

Back Propagation error correction at MLP node

Partial differentiation of Eq.3.2.1 w.r.t y_{in}

$$\frac{\partial y}{\partial y_{in}} = \partial fastsigmoid(y_{in})$$

where Chapter 1.0.1 delineates $\partial fastsigmoid()$ in detail

$$\partial y_{in} = \partial y \cdot \partial fastsigmoid(y_{in}) \tag{3.2.2}$$

$$y_{in} = W \times X + b \tag{3.2.3}$$

Partial differentiation of Eq.3.2.3 w.r.t W

$$\frac{\partial y_{in}}{\partial W} = X$$

$$\begin{aligned} \partial W &= \partial y_{in} \times X' \\ &= \partial y \cdot \partial fastsigmoid(y_{in}) \times X' \end{aligned} \tag{3.2.4}$$

Partial differentiation of Eq.3.2.3 w.r.t b

$$\frac{\partial y_{in}}{\partial b} = 1$$

$$\begin{aligned} \partial b &= \partial y_{in} \\ &= \partial y \cdot \partial fastsigmoid(y_{in}) \end{aligned} \tag{3.2.5}$$

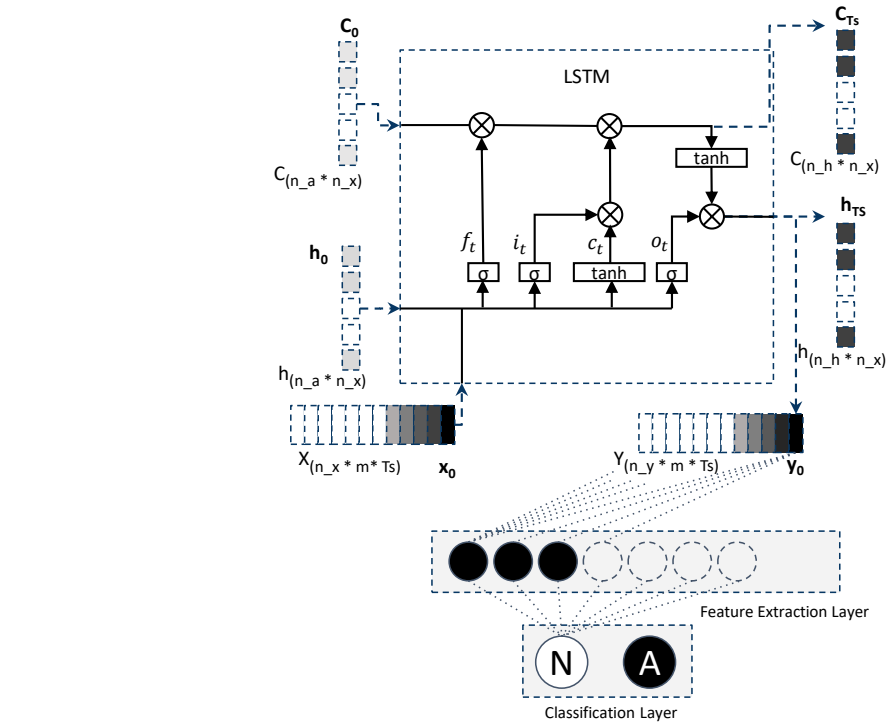
Back Propagation error to the previous layer

Partial differentiation of Eq.3.2.3 w.r.t X

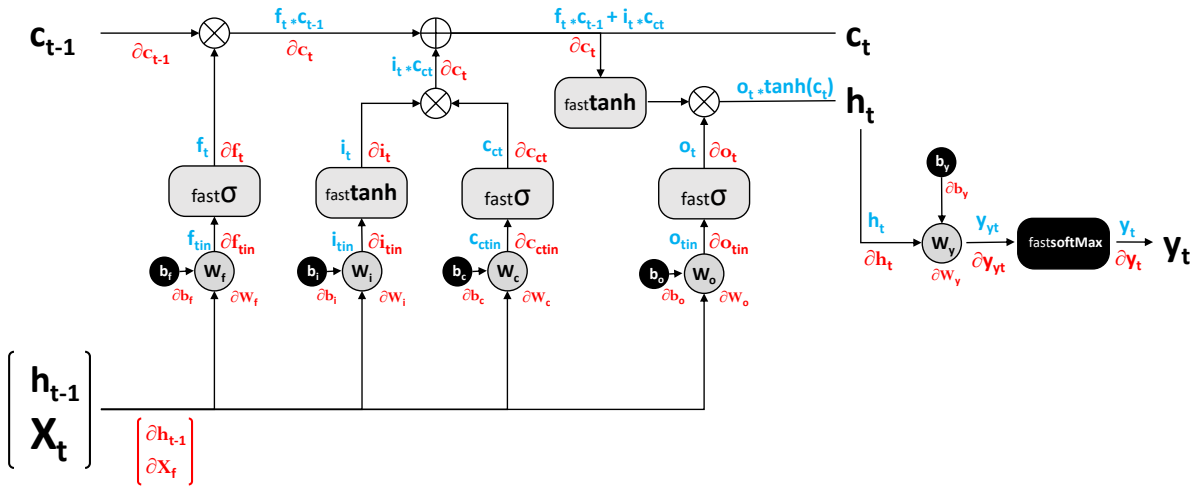
$$\frac{\partial y_{in}}{\partial X} = W$$

$$\begin{aligned} \partial X &= W' \times \partial y_{in} \\ &= W' \times (\partial y \cdot \partial fastsigmoid(y_{in})) \end{aligned} \tag{3.2.6}$$

2.2.2 LSTM Cell



(a) LSTM Cell Pipeline



(b) Forward and Back Propagation through single time step at LSTM Cell

Figure 2.4: Long Short Term Memory in Fixed Point Environment

n_h	: Size of Hidden State vector of single LSTM cell
n_f (default n_f = 1)	: Size of One hot vector representation of element in input vector X
n_x	: Size of Input vector X
$fast\sigma / fastsigmoid()$	: Fast approximation of Sigmoid activation function (1.0.1)
$fast\tanh()$	: Fast approximation of Tanh activation function (1.0.1)
$fastsoftMax()$	: Fast approximation of SoftMax function (1.0.1)

Forward Prediction

Forget gate f_t:

$$f_{t_{[n_h \times n_f]}} = fast\sigma(W_{f_{[n_h \times (n_h+n_x)]}} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}_{[(n_h+n_x) \times n_f]} + b_{f_{[n_h \times n_f]}}) \tag{4.1.1}$$

Input gate i_t:

$$i_{t_{[n_h \times n_f]}} = fast\sigma(W_{i_{[n_h \times (n_h+n_x)]}} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}_{[(n_h+n_x) \times n_f]} + b_{i_{[n_h \times n_f]}}) \tag{4.1.2}$$

Candidate gate c_{ct}:

$$c_{ct_{[n_h \times n_f]}} = fast\sigma(W_{c_{[n_h \times (n_h+n_x)]}} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}_{[(n_h+n_x) \times n_f]} + b_{c_{[n_h \times n_f]}}) \tag{4.1.3}$$

Cell state c_t:

$$c_{t_{[n_h \times n_f]}} = f_{t_{[n_h \times n_f]}} \times c_{t-1_{[n_h \times n_f]}} + i_{t_{[n_h \times n_f]}} \times c_{ct_{[n_h \times n_f]}} \tag{4.1.4}$$

Output gate o_t:

$$o_{t_{[n_h \times n_f]}} = fast\sigma(W_{o_{[n_h \times (n_h+n_x)]}} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}_{[(n_h+n_x) \times n_f]} + b_{o_{[n_h \times n_f]}}) \tag{4.1.5}$$

Hidden state h_t:

$$h_{t_{[n_h \times n_f]}} = o_{t_{[n_h \times n_f]}} \times fast\tanh(c_t)_{[n_h \times n_f]} \tag{4.1.6}$$

Output y_t:

$$y_{yt_{[n_y \times 1]}} = W_{y_{[n_y \times n_h]}} \times h_{t_{[n_h \times 1]}} + b_{y_{[n_y \times 1]}} \tag{4.1.7}$$

$$y_{t_{[n_y \times 1]}} = fastsoftMax(y_{yt_{[n_y \times 1]}}) \tag{4.1.8}$$

Back Propagation

$$y_t = fastsoftMax(y_{yt}) \tag{4.2.1}$$

Back Propagation error correction at Output (y)

Partial differentiation of Eq.4.2.1 w.r.t y_{yt}

$$\frac{\partial y_t}{\partial y_{yt}} = \frac{I}{I + y_{yt}} \cdot y_t \cdot (I - y_t)$$

$$\partial y_{yt} = \partial y_t \cdot \frac{I}{I + y_{yt}} \cdot y_t \cdot (I - y_t) \tag{4.2.2}$$

$$y_{yt} = W_y \times h_t + b_y \tag{4.2.3}$$

Partial differentiation of Eq.4.2.3 w.r.t W_y

$$\frac{\partial y_{yt}}{\partial W_y} = h_t$$

$$\begin{aligned} \partial W_y &= \partial y_{yt} \times h'_t \\ &= (\partial y_t \cdot \frac{I}{I + y_{yt}} \cdot y_t \cdot (I - y_t)) \times h'_t \end{aligned} \tag{4.2.4}$$

Partial differentiation of Eq.4.2.3 w.r.t b_y

$$\frac{\partial y_{yt}}{\partial b_y} = 1$$

$$\begin{aligned} \partial b_y &= \partial y_{yt} \\ &= \partial y_t \cdot \frac{I}{I + y_{yt}} \cdot y_t \cdot (I - y_t) \end{aligned} \tag{4.2.5}$$

Back Propagation error in Hidden State (h)

Partial differentiation of Eq.4.2.3 w.r.t h_t

$$\frac{\partial y_{yt}}{\partial h_t} = W_y$$

$$\partial h_t = W'_y \times \partial y_{yt} + \delta h_t$$

where $\delta h_t = 0$ for final time step (Ts) and ∂h_{t+1} for the rest

$$\partial h_t = W'_y \times (\partial y_t \cdot \frac{I}{I + y_{yt}} \cdot y_t \cdot (I - y_t)) + \delta h_t \tag{4.2.6}$$

$$h_t = o_t \cdot fast\tanh(c_t) \tag{4.2.7}$$
$$\text{where } o_t = fastsigmoid(o_{tin})$$
$$c_t = fast\tanh(c_{tin})$$

Back Propagation error correction at Output Gate (o)

Partial differentiation of Eq.4.2.7 w.r.t o_{tin}

$$\frac{\partial h_t}{\partial o_{t_{in}}} = \partial fastsigmoid(o_{tin}) \cdot fast\tanh(c_t)$$

$$\partial o_{t_{in}} = \partial h_t \cdot \partial fastsigmoid(o_{tin}) \cdot fast\tanh(c_t) \tag{4.2.8}$$

$$o_{tin} = W_o \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix} + b_o \tag{4.2.9}$$

Partial differentiation of Eq.4.2.9 w.r.t W_o

$$\frac{\partial o_{tin}}{\partial W_o} = \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}$$

$$\begin{aligned} \partial W_o &= \partial o_{tin} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}' \\ &= (\partial h_t \cdot \partial fastsigmoid(o_{tin}) \cdot fast\tanh(c_t)) \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}' \\ &= ((W_y' \times (\partial y_t \cdot \frac{I}{I + y_{yt}} \cdot y_t \cdot (I - y_t))) \cdot \partial fastsigmoid(o_{tin}) \cdot fast\tanh(c_t)) \\ &\qquad \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}' \end{aligned} \tag{4.2.10}$$

Partial differentiation of Eq.4.2.9 w.r.t b_o

$$\frac{\partial o_{tin}}{\partial b_o} = 1$$

$$\begin{aligned} \partial b_o &= \partial o_{tin} \\ &= (\partial h_t \cdot \partial fastsigmoid(o_{tin}) \cdot fast\tanh(c_t)) \\ &= ((W_y' \times (\partial y_t \cdot \frac{I}{I + y_{yt}} \cdot y_t \cdot (I - y_t))) \cdot \partial fastsigmoid(o_{tin}) \cdot fast\tanh(c_t)) \end{aligned} \tag{4.2.11}$$

Back Propagation error in Cell State (c)

Partial differentiation of Eq.4.2.7 w.r.t c_t

$$\begin{aligned}\frac{\partial h_t}{\partial c_t} &= o_t \cdot \partial fast\tanh(c_t)\\ &\text{where Chapter 1.0.1 delineates } \partial fast\tanh() \text{ in detail}\end{aligned}$$

$$\begin{aligned}\partial c_t &= \partial h_t \cdot o_t \cdot \partial fast\tanh(c_t) + \delta c_t \\ &\text{where } \delta c_t = 0 \text{ for final time step } (Ts) \text{ and } \partial c_{t+1} \text{ for the rest}\end{aligned} \tag{4.2.12}$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot c_{ct} \tag{4.2.13}$$

Back Propagation error correction at Candidate Gate (c_c)

Partial differentiation of Eq.4.2.13 w.r.t c_{ctin}

$$\begin{aligned}\frac{\partial c_t}{\partial c_{ctin}} &= i_t \cdot \frac{\partial fast\tanh(c_{ctin})}{\partial c_{ctin}}\\ &= i_t \cdot \partial fast\tanh(c_{ctin})\end{aligned}$$

$$\partial c_{ctin} = \partial c_t \cdot i_t \cdot \partial fast\tanh(c_{ctin}) \tag{4.2.14}$$

$$c_{ctin} = W_c \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix} + b_c \tag{4.2.15}$$

Partial differentiation of Eq.4.2.15 w.r.t W_c

$$\begin{aligned}\partial W_c &= \partial c_{ctin} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}'\\ &= (\partial c_t \cdot i_t \cdot \partial fast\tanh(c_{ctin}) \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}'\end{aligned} \tag{4.2.16}$$

Partial differentiation of Eq.4.2.15 w.r.t b_c

$$\begin{aligned}\partial b_c &= \partial c_{ctin}\\ &= \partial c_t \cdot i_t \cdot \partial fast\tanh(c_{ctin})\end{aligned} \tag{4.2.17}$$

Back Propagation error correction at Input Gate (i)

Partial differentiation of Eq.4.2.13 w.r.t i_{tin}

$$\begin{aligned}\frac{\partial c_t}{\partial i_{tin}} &= \frac{\partial fastsigmoid(i_{tin})}{\partial i_{tin}} \cdot c_{ct} \\ &= \partial fastsigmoid(i_{tin}) \cdot c_{ct}\end{aligned}$$

$$\partial i_{tin} = \partial c_t \cdot \partial fastsigmoid(i_{tin}) \cdot c_{ct} \tag{4.2.18}$$

$$i_{tin} = W_i \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix} + b_i \tag{4.2.19}$$

Partial differentiation of Eq.4.2.19 w.r.t W_i

$$\frac{\partial i_{tin}}{\partial W_i} = \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}$$

$$\begin{aligned}\partial W_i &= \partial i_{tin} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}' \\ &= \partial c_t \cdot \partial fastsigmoid(i_{tin}) \cdot c_{ct} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}'\end{aligned} \tag{4.2.20}$$

Partial differentiation of Eq.4.2.19 w.r.t b_i

$$\frac{\partial i_{tin}}{\partial b_i} = 1$$

$$\begin{aligned}\partial b_i &= \partial i_{tin} \\ &= \partial c_t \cdot \partial fastsigmoid(i_{tin}) \cdot c_{ct}\end{aligned} \tag{4.2.21}$$

Back Propagation error correction at Forget Gate (f)

Partial differentiation of Eq.4.2.13 w.r.t f_{tin}

$$\begin{aligned}\frac{\partial c_t}{\partial f_{tin}} &= \frac{\partial fastsigmoid(f_{tin})}{\partial f_{tin}} \cdot c_{t-1} \\ &= \partial fastsigmoid(f_{tin}) \cdot c_{t-1}\end{aligned}$$

$$\partial f_{tin} = \partial c_t \cdot \partial fastsigmoid(f_{tin}) \cdot c_{t-1} \tag{4.2.22}$$

$$f_{tin} = W_f \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix} + b_f \tag{4.2.23}$$

Partial differentiation of Eq.4.2.23 w.r.t W_f

$$\frac{\partial f_{tin}}{\partial W_f} = \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}$$

$$\begin{aligned} \partial W_f &= \partial f_{tin} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}' \\ &= \partial c_t \cdot \partial fastsigmoid(f_{tin}) \cdot c_{t-1} \times \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}' \end{aligned} \tag{4.2.24}$$

Partial differentiation of Eq.4.2.23 w.r.t b_f

$$\frac{\partial f_{tin}}{\partial b_f} = 1$$

$$\begin{aligned} \partial b_f &= \partial f_{tin} \\ &= \partial c_t \cdot \partial fastsigmoid(f_{tin}) \cdot c_{t-1} \end{aligned} \tag{4.2.25}$$

Back Propagation error correction at h_{t-1}

$$\frac{\partial \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix}}{\partial W} = \partial f_{tin} + \partial i_{tin} + \partial c_{ctin} + \partial o_{tin} \tag{4.2.26}$$

$$\partial \begin{bmatrix} h_{t-1} \\ X_t \end{bmatrix} = W_f' \times \partial f_{tin} + W_i' \times \partial i_{tin} + W_c' \times \partial c_{ctin} + W_o' \times \partial o_{tin} \tag{4.2.27}$$

Back Propagation error correction at c_{t-1}

$$\frac{\partial c_t}{\partial c_{t-1}} = f_t \tag{4.2.28}$$

$$\partial c_{t-1} = \partial c_t \cdot f_t \tag{4.2.29}$$