

AILAB SEMINAR #19

RNN & LSTM

220209 한양대학교 인공지능연구실 김민지

1

DNN

DEEP NEURAL NETWORK

DNN - Deep Neural Network

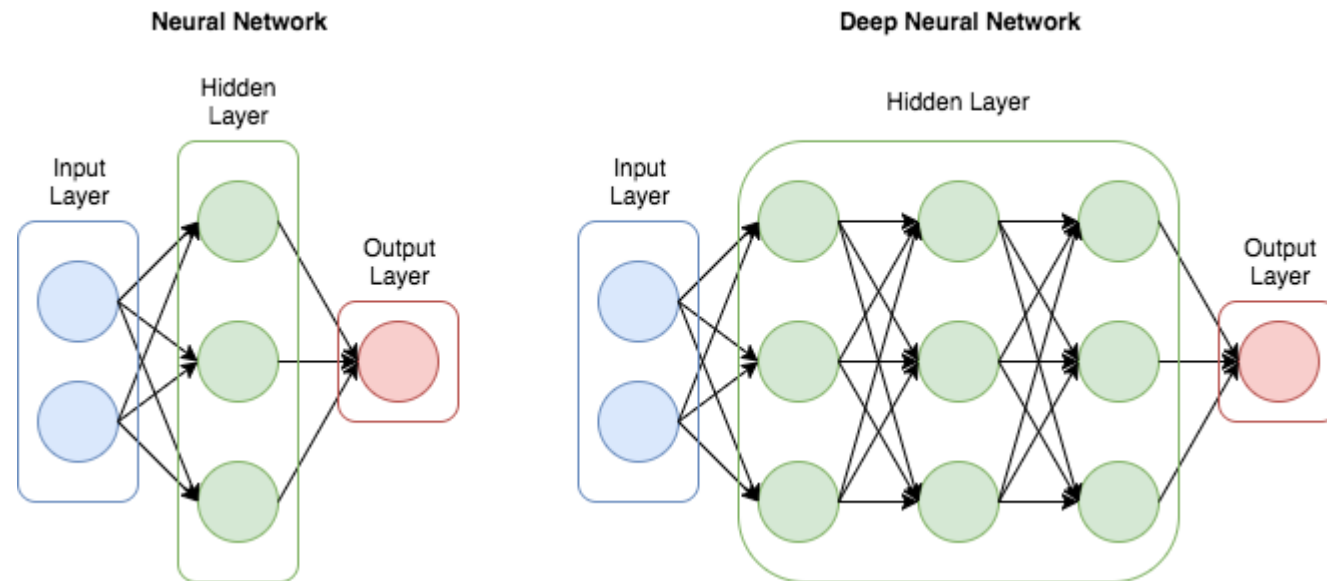
DNN

RNN

LSTM

GRU

- Input layer 와 Output layer 사이 여러 개의 Hidden layer 가 있는 인공 신경망의 한 형태
- Feed-Forward 와 Backpropagation 을 통해 Loss를 낮추는 방향으로 Parameter를 학습



<https://ebbnflow.tistory.com/119>

DNN

RNN

LSTM

GRU

- 고정된 길이의 Input과 Output에 대해서만 연산이 가능
- 시계열, 자연어 등 Sequence data에서 Context 정보를 활용하기 어려움

자연어

This sentence is a sequence of words...



주가



음성



2

RNN

RECURRENT NEURAL NETWORK

RNN - Recurrent Neural Network

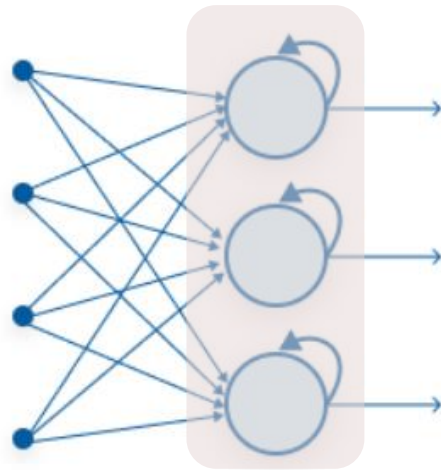
DNN

RNN

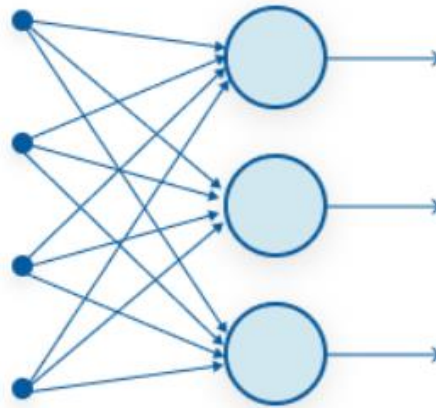
LSTM

GRU

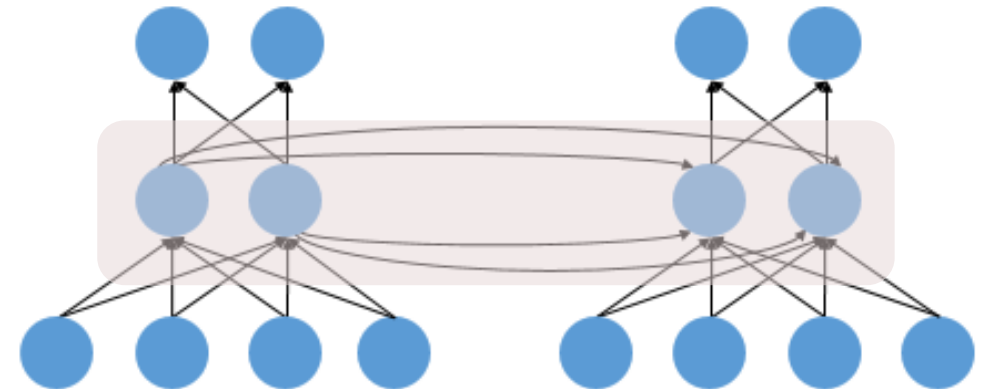
- Hidden layer의 결과 값을 다음 시점의 Hidden Layer로 보내는 재귀적 특성을 가진 순환 신경망



Recurrent Neural Network



Feed-Forward Neural Network

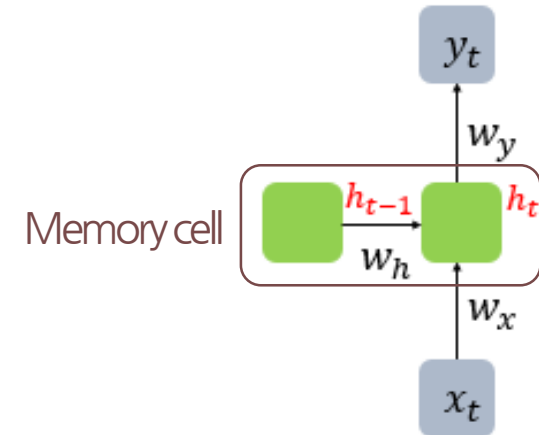
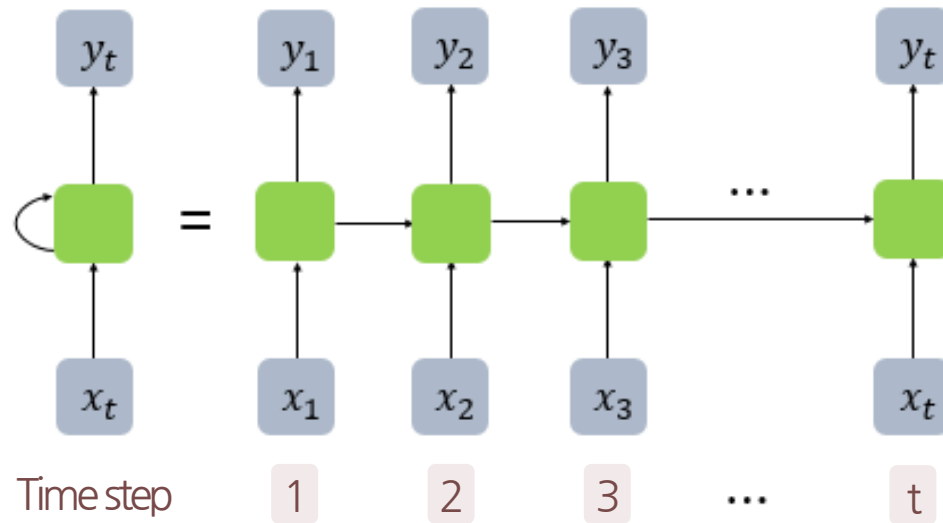


Time step t

Time step $t+1$

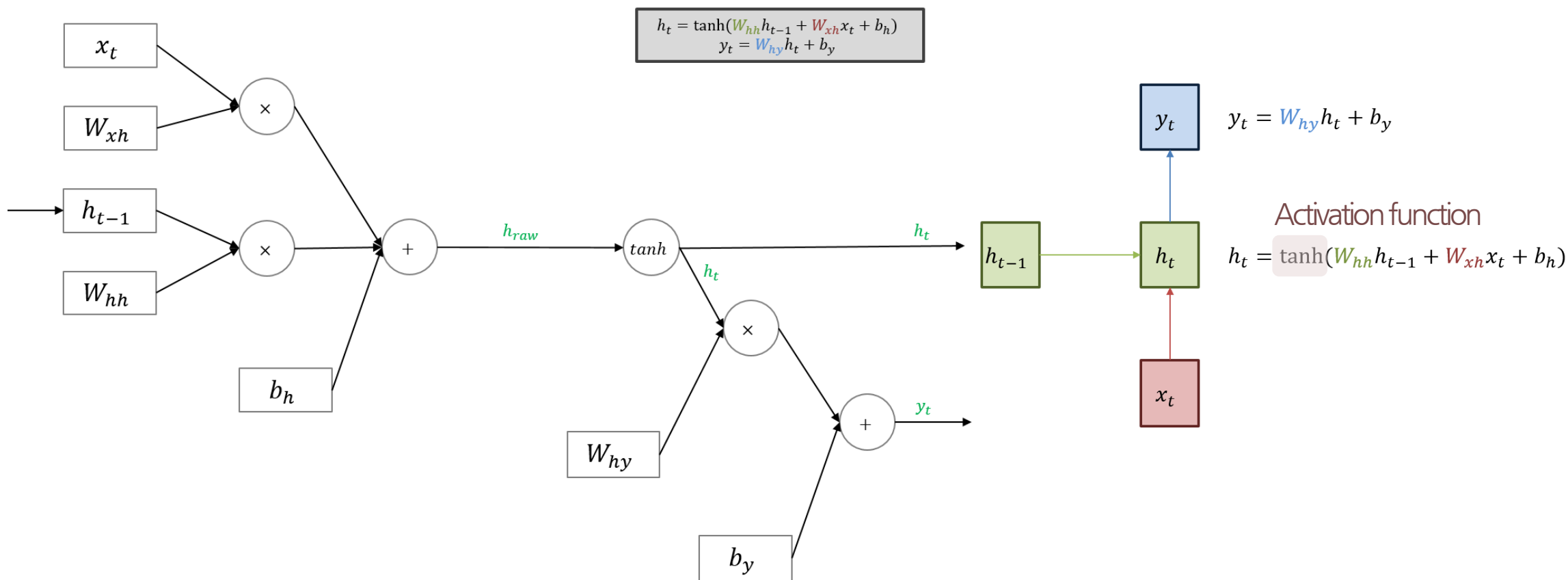
RNN - Recurrent Neural Network

- Memory cell (RNN cell) 은 이전의 값(Hidden state)을 기억하여 Sequential 정보를 학습



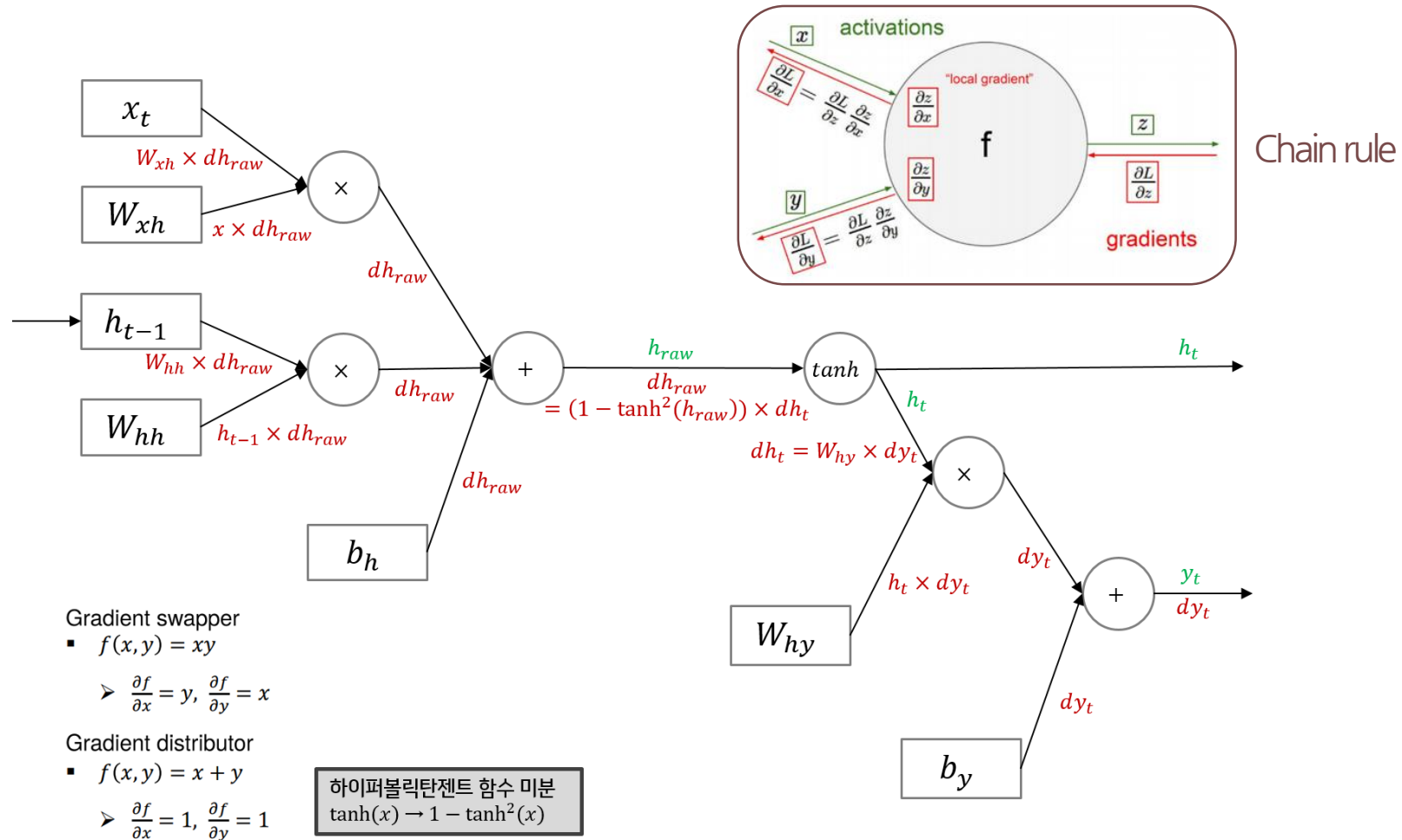
RNN - The Vanilla RNN Forward

- 시점 t에서 Input = x_t , Hidden state = h_t , Output = y_t
- 각 시점의 W_{hy} , W_{hh} , W_{xh} 값을 공유함



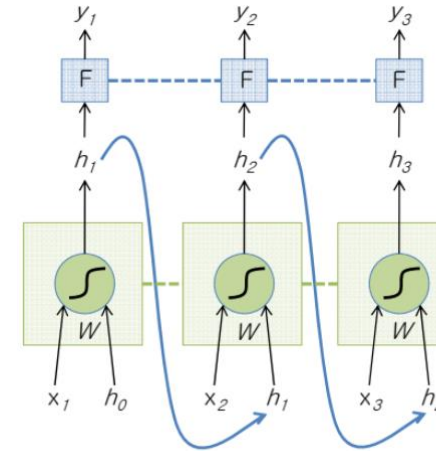
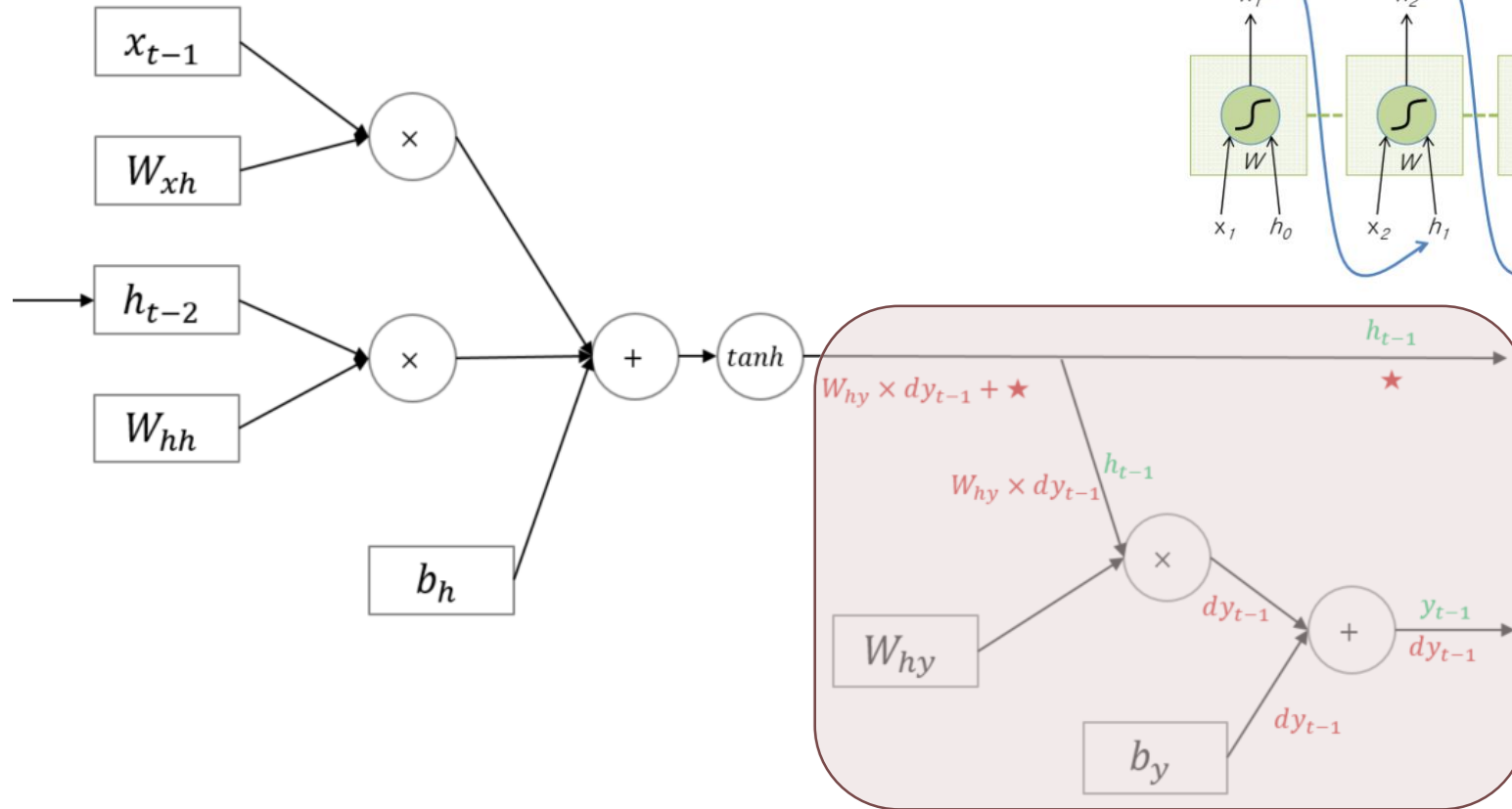
RNN - Backpropagation Through Time (BPTT)

- Time step에 따라 Unfold 된 RNN Cell을 DNN과 유사하게 Backpropagation



RNN - Backpropagation Through Time (BPTT)

- 순환 구조이므로 다음 시점의 Gradient도 더해서 반영함



$$h_t = \tanh W \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix}$$

$$y_t = F(h_t)$$

$$C_t = \text{Loss}(y_t, \text{GT}_t)$$

----- indicates shared weights

모든 경로의 derivative를 곱해서 더함

- Input과 Output의 길이가 가변적이므로 다양한 활용이 가능

one to one

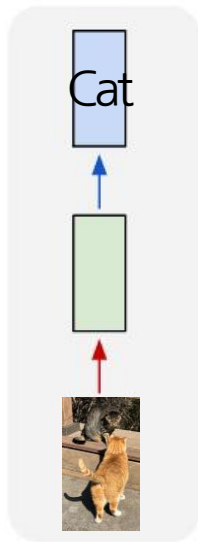


image
classification

one to many

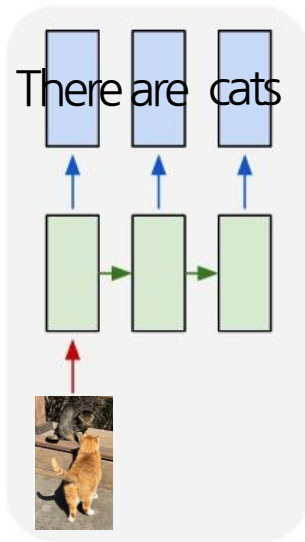
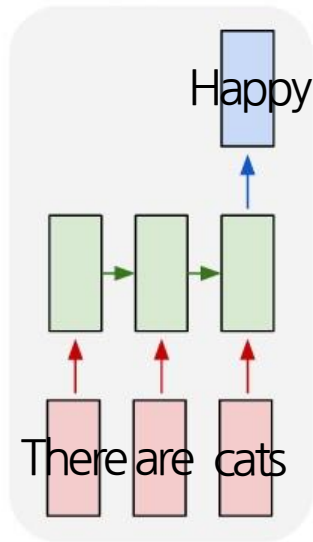


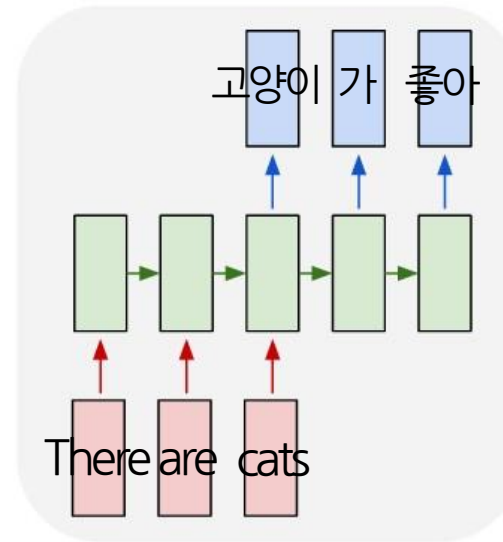
Image
captioning

many to one



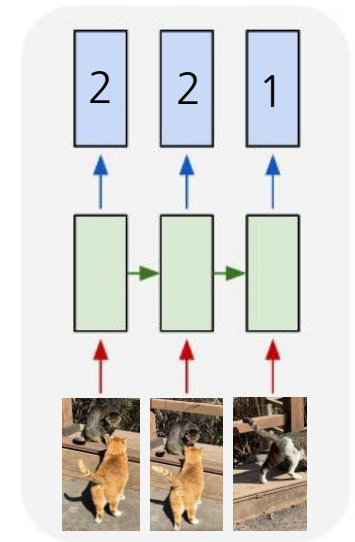
Sentimental
analysis

many to many



Machine
translation

many to many



Video
Classification

DNN

RNN

LSTM

GRU

- Example - Minimal character-level language model (The Little Prince)

90547 Characters, 80 Unique

```
----  
!PweW-4'!2y7r YuD3Ev"sSefyj$KHW"sP"T9EUAYNq)BT2WaHBjpporE7rCm.D zanzbchlj$h3,ek-b0'yKpG-Tx5"5Exh fz-jVi$$Z'U'0xh)FI LRh"w0G  
hW s8'TFHg1f"arg5LV"p9AZ.q.NBvrr"fj0!4ft-3(A""g"eeB:illE'y!Nb\spUF7kk')oL-!\n  
----  
iter 0, loss: 109.550664
```



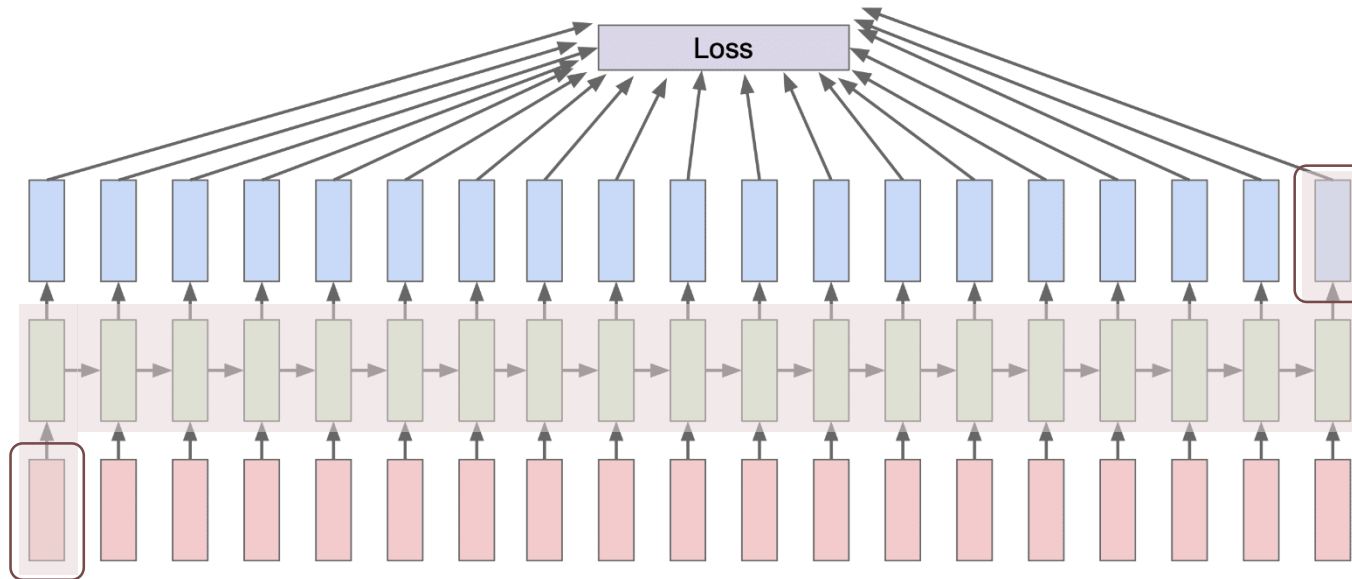
```
----  
t deed seece,..  
  
"I  
hom, insed tasly.  
  
"0ner, is il le ind sas yowe the junco hame the was it are. kne ad at That ang they wn of a ctaide beake.  
  
"Thay laosy shao ave revker ast foubbet if a cee ghec  
----  
iter 10000, loss: 54.261064
```



```
----  
n," the long muke you dot shimpsik to is waughe thowen'twambabge the was mollije, coad them id as tarning  
you stime fout tho kill, my wass. Mins," serest-ming to he and whey to he aldile, in you corsp  
----  
iter 50000, loss: 41.201140
```

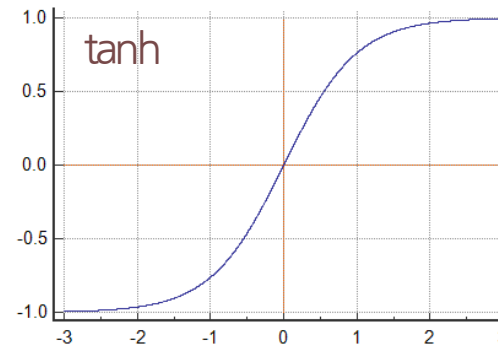
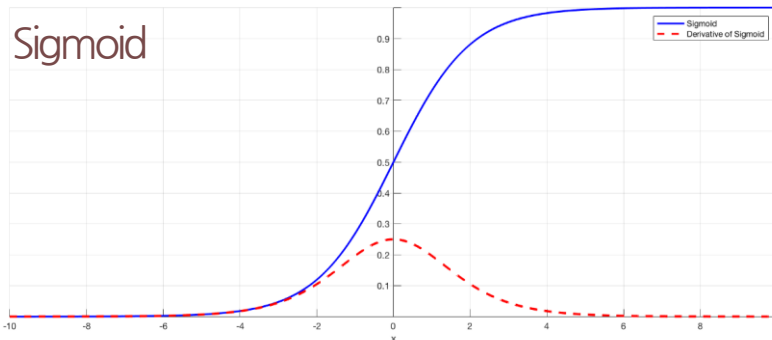
RNN - BPTT의 한계

- 모든 Time step마다 Unfold된 RNN Cell의 처음부터 끝까지 연산해야 함



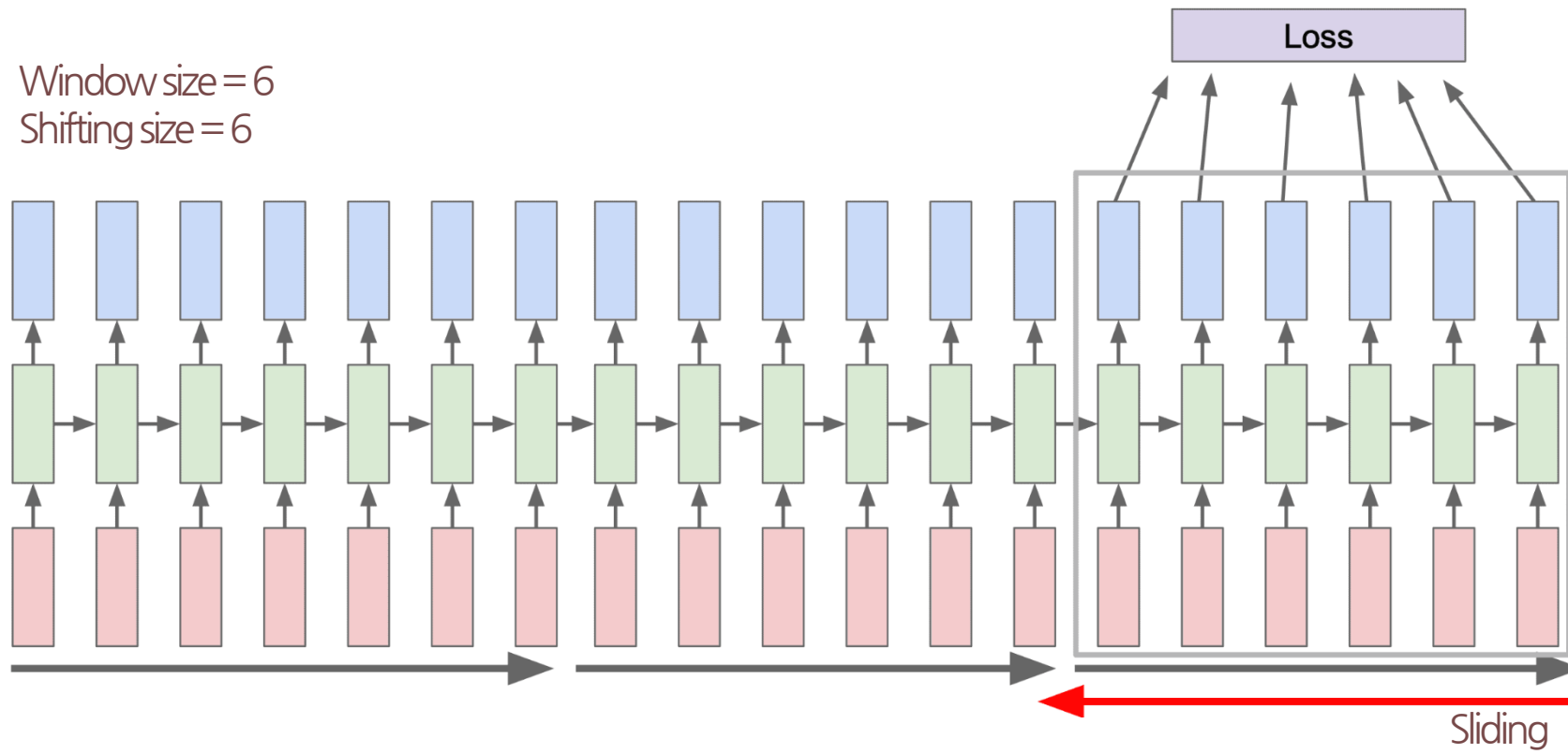
Gradient Clipping (Heuristic)

- Gradients vanishing(or exploding) problem
- Memory에 저장해야 하는 값이 많음
- 연산이 많아서 학습 시간이 오래 걸림



RNN - Truncated BPTT

- Time step을 일정 크기로 나누어서 Backpropagation
- True Truncated BPTT는 Shifting size = 1인 경우를 의미



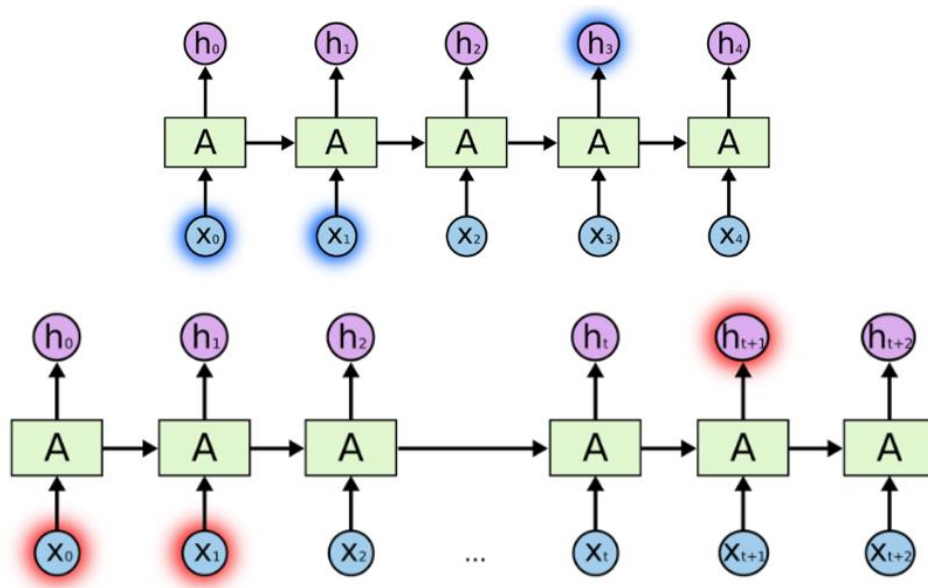
DNN

RNN

LSTM

GRU

- 앞선 방법으로는 Long-term dependency problem이 완전히 해결되지 않음
- 거리가 먼 정보는 반영되기 어려워 학습 능력이 저하됨



$$\frac{\partial C_t}{\partial h_1} = \left(\frac{\partial C_t}{\partial y_t} \right) \left(\frac{\partial y_t}{\partial h_1} \right)$$

$$= \left(\frac{\partial C_t}{\partial y_t} \right) \left(\frac{\partial y_t}{\partial h_t} \right) \left(\frac{\partial h_t}{\partial h_{t-1}} \right) \dots \left(\frac{\partial h_2}{\partial h_1} \right)$$

Constant Error Flow 필요

$$h_t = \tanh W \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix}$$

$$y_t = F(h_t)$$

$$C_t = \text{Loss}(y_t, GT_t)$$

3

LSTM

LONG SHORT-TERM MEMORY

LSTM - Long Short-Term Memory

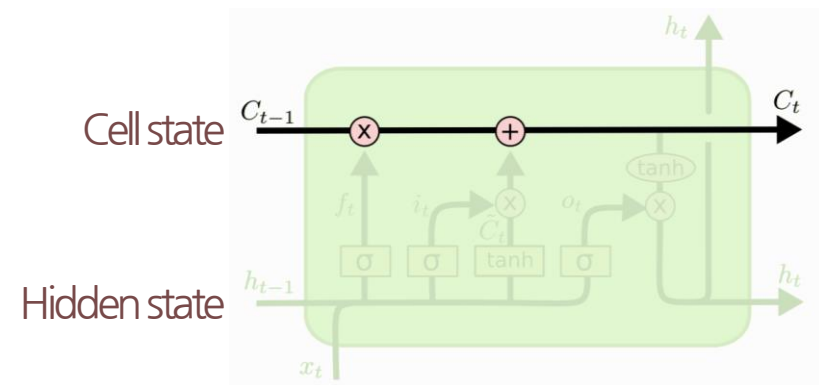
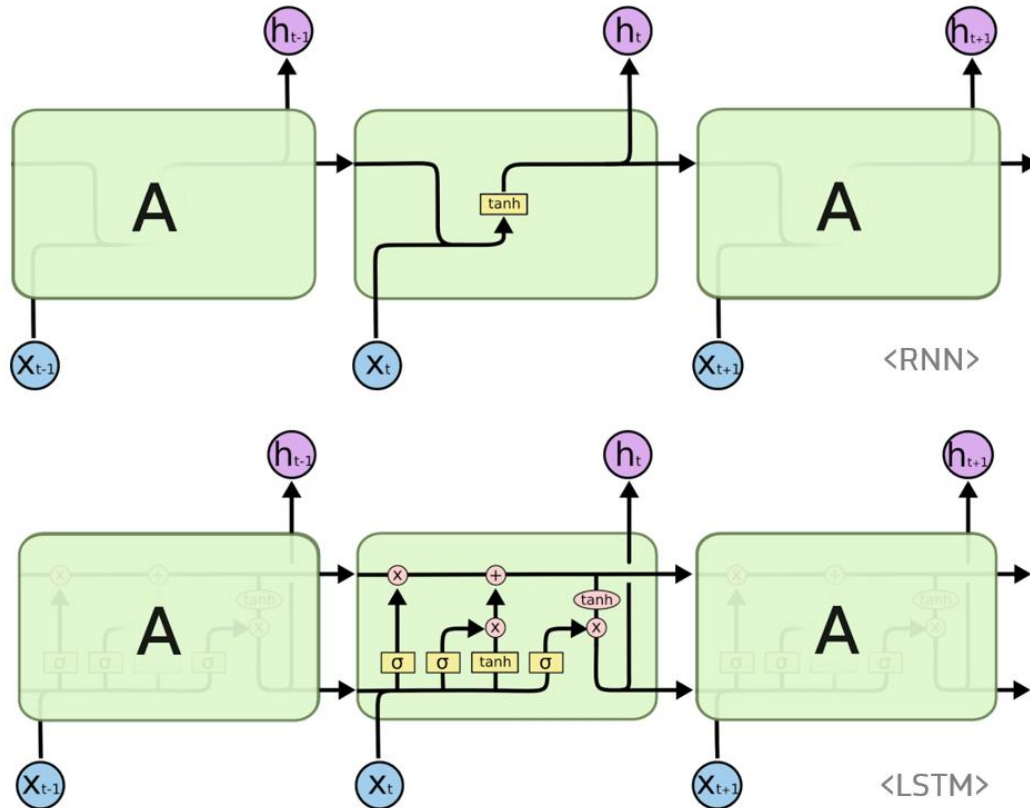
DNN

RNN

LSTM

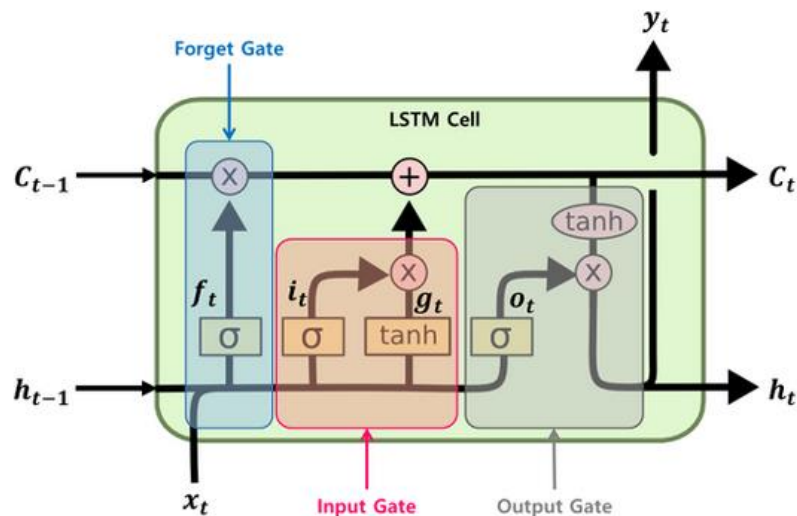
GRU

- 단기 기억을 위한 Hidden state 이외에 장기 기억을 위한 Cell state를 도입

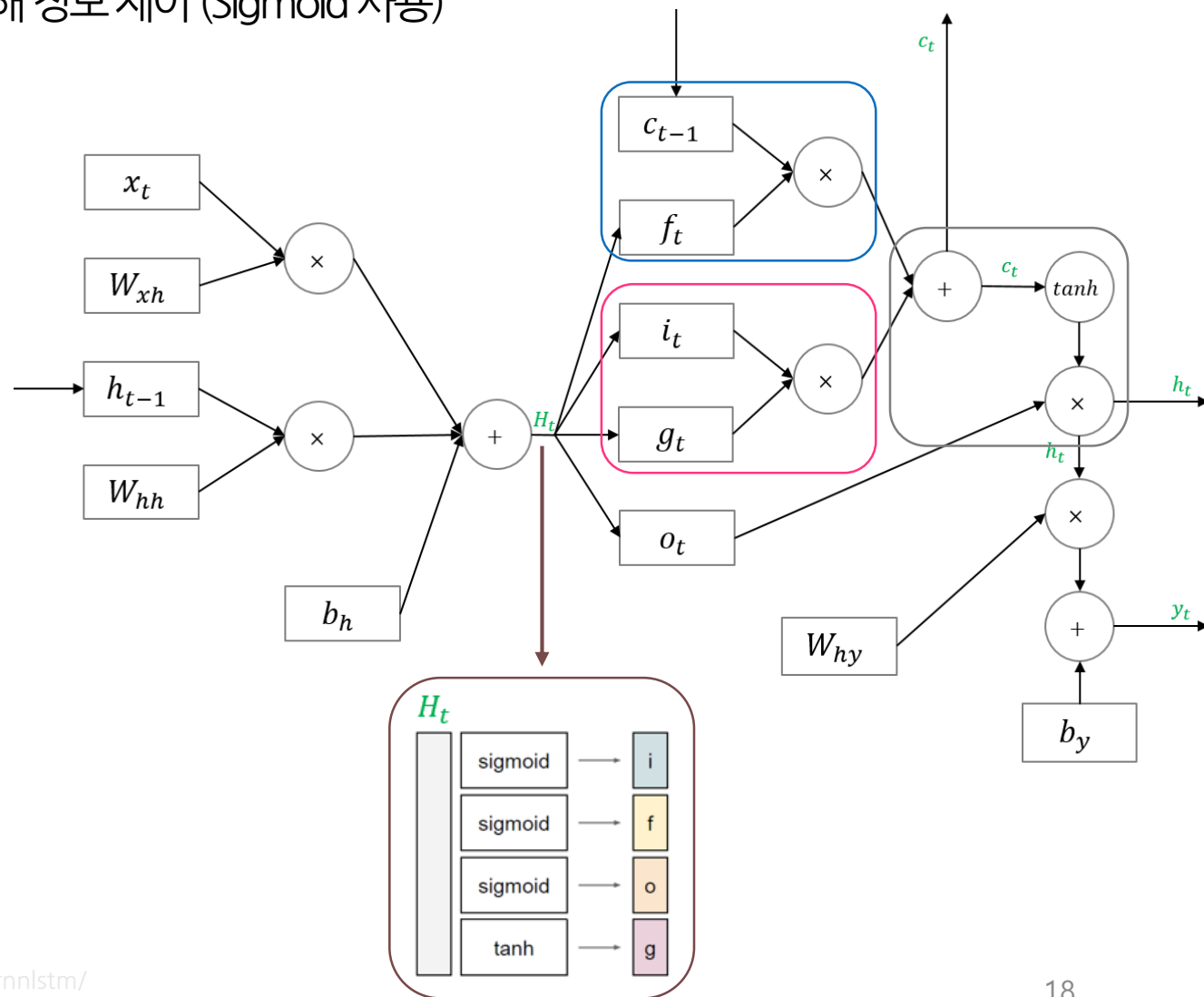


LSTM - LSTM Cell Forward Pass

- Forget gate, input gate, output gate를 통해 정보 제어 (Sigmoid 사용)



$$\begin{aligned}
 f_t &= \sigma(W_{xh_f}x_t + W_{hh_f}h_{t-1} + b_{h_f}) \\
 i_t &= \sigma(W_{xh_i}x_t + W_{hh_i}h_{t-1} + b_{h_i}) \\
 o_t &= \sigma(W_{xh_o}x_t + W_{hh_o}h_{t-1} + b_{h_o}) \\
 g_t &= \tanh(W_{xh_g}x_t + W_{hh_g}h_{t-1} + b_{h_g}) \\
 c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
 h_t &= o_t \odot \tanh(c_t)
 \end{aligned}$$



LSTM - LSTM Forget gate layer / Input gate layer

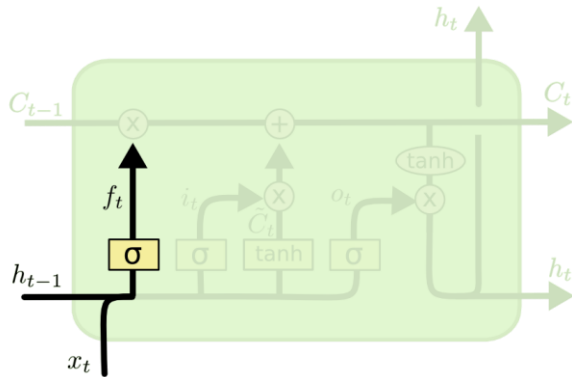
DNN

RNN

LSTM

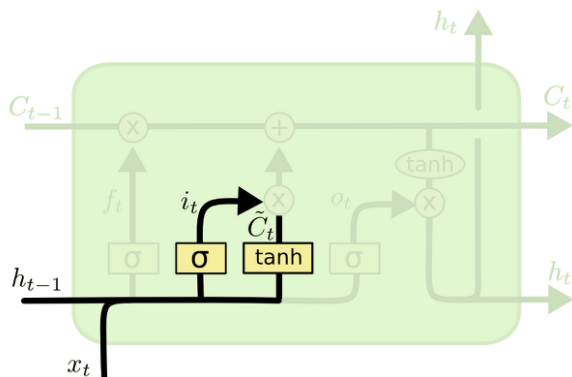
GRU

- Forget gate - Cell state로부터 어떤 정보를 잊을 것인가



$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

- Input gate - 새로운 정보 중 어떤 것을 Cell state에 저장할 것인가



$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

LSTM - LSTM Cell state update / Output gate layer

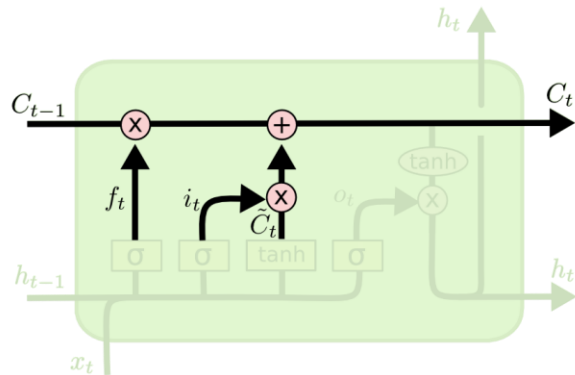
DNN

RNN

LSTM

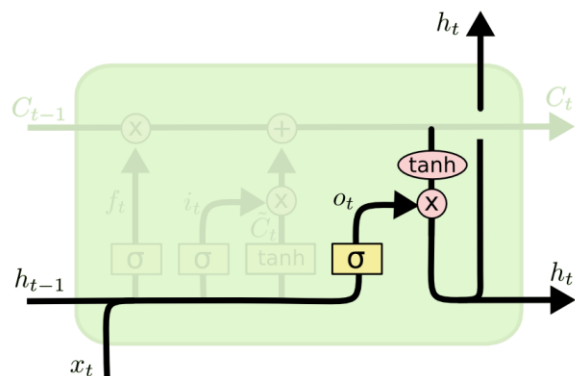
GRU

- Cell state update - Forget gate와 Input gate를 적용



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

- Output gate - Cell state 중 어느 부분을 Output으로 내보낼 것인가

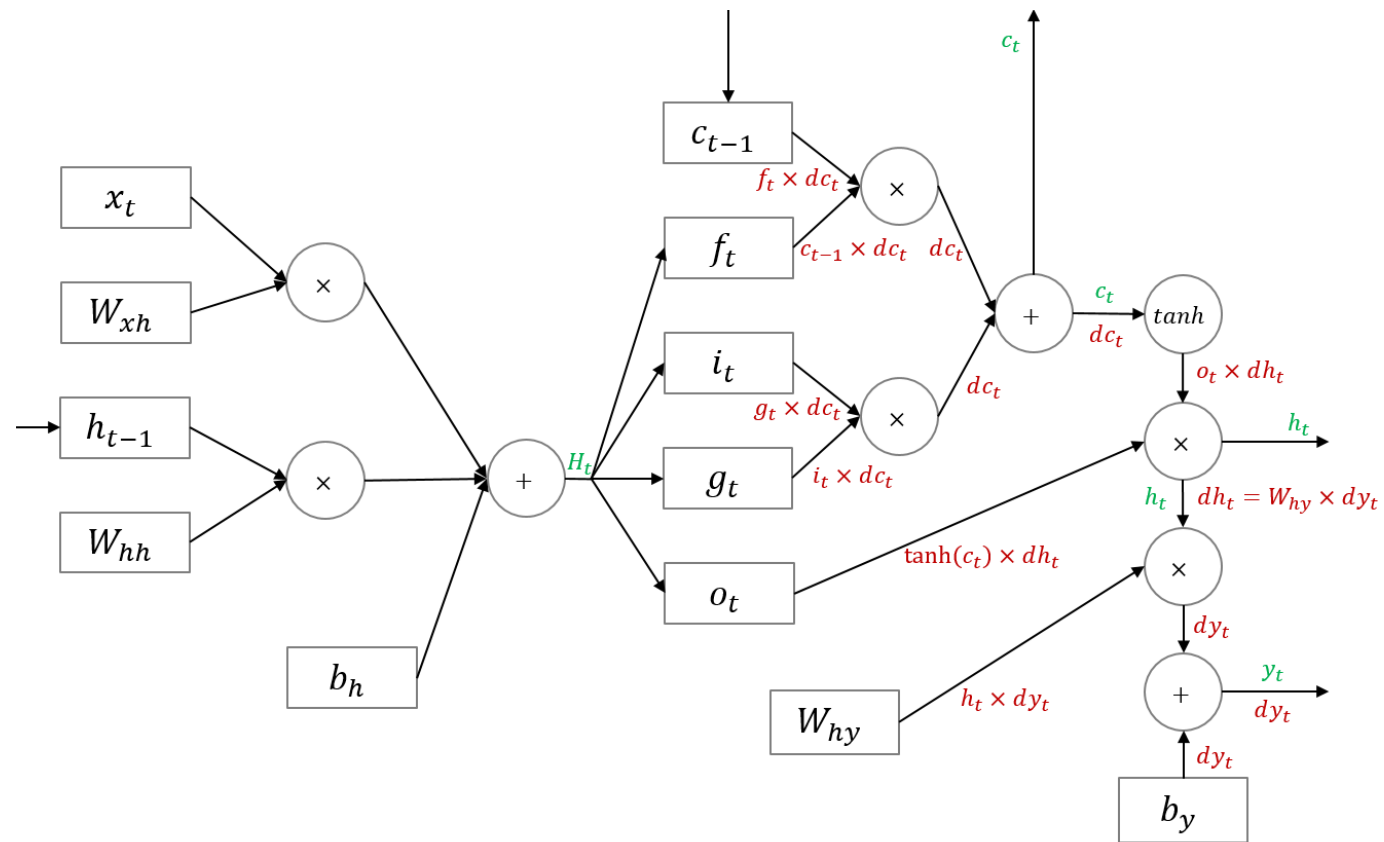


$$o_t = \sigma(W_o [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * \tanh(C_t)$$

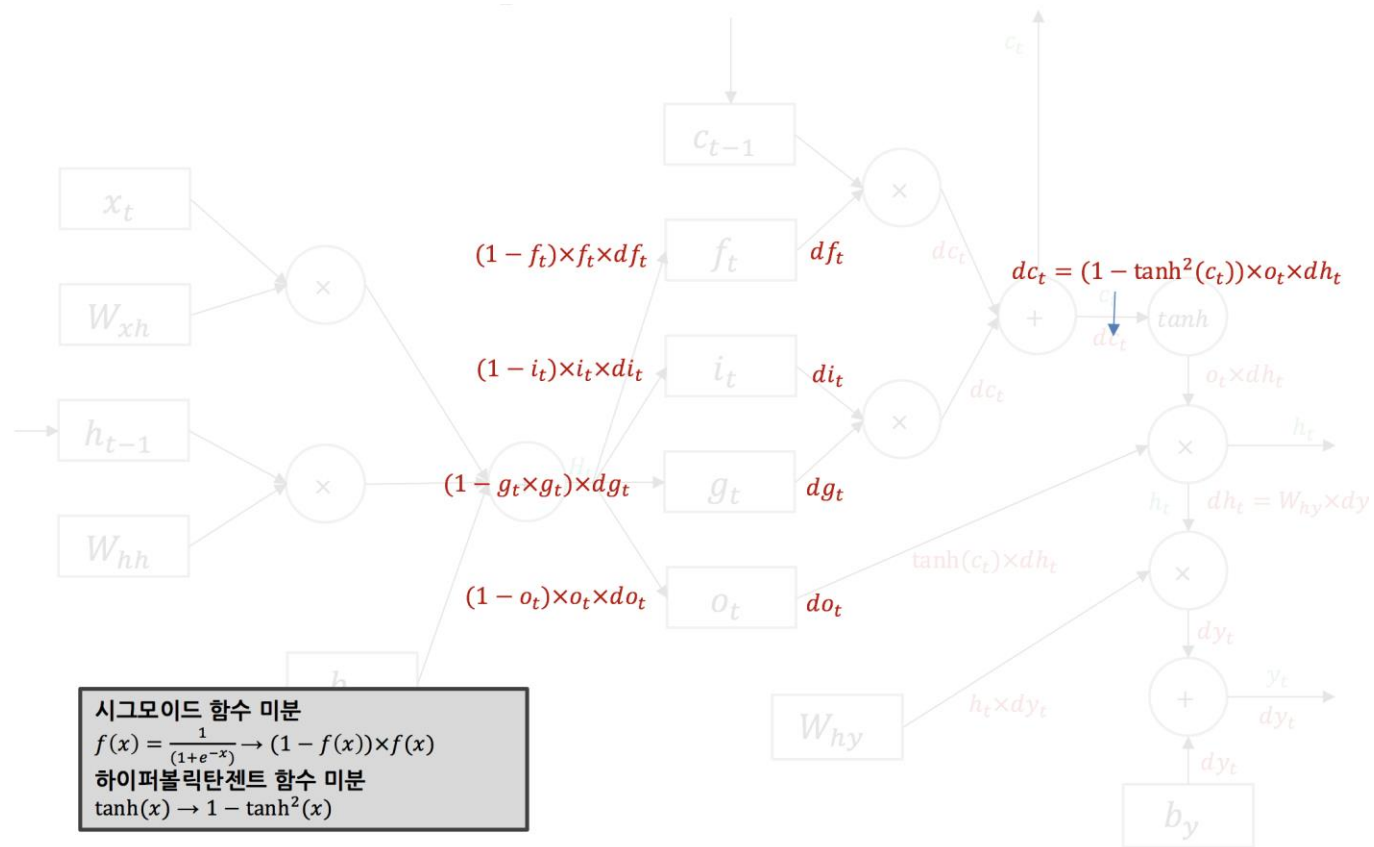
LSTM - LSTM Cell Backpropagation

- RNN 에서의 Backpropagation 과 유사



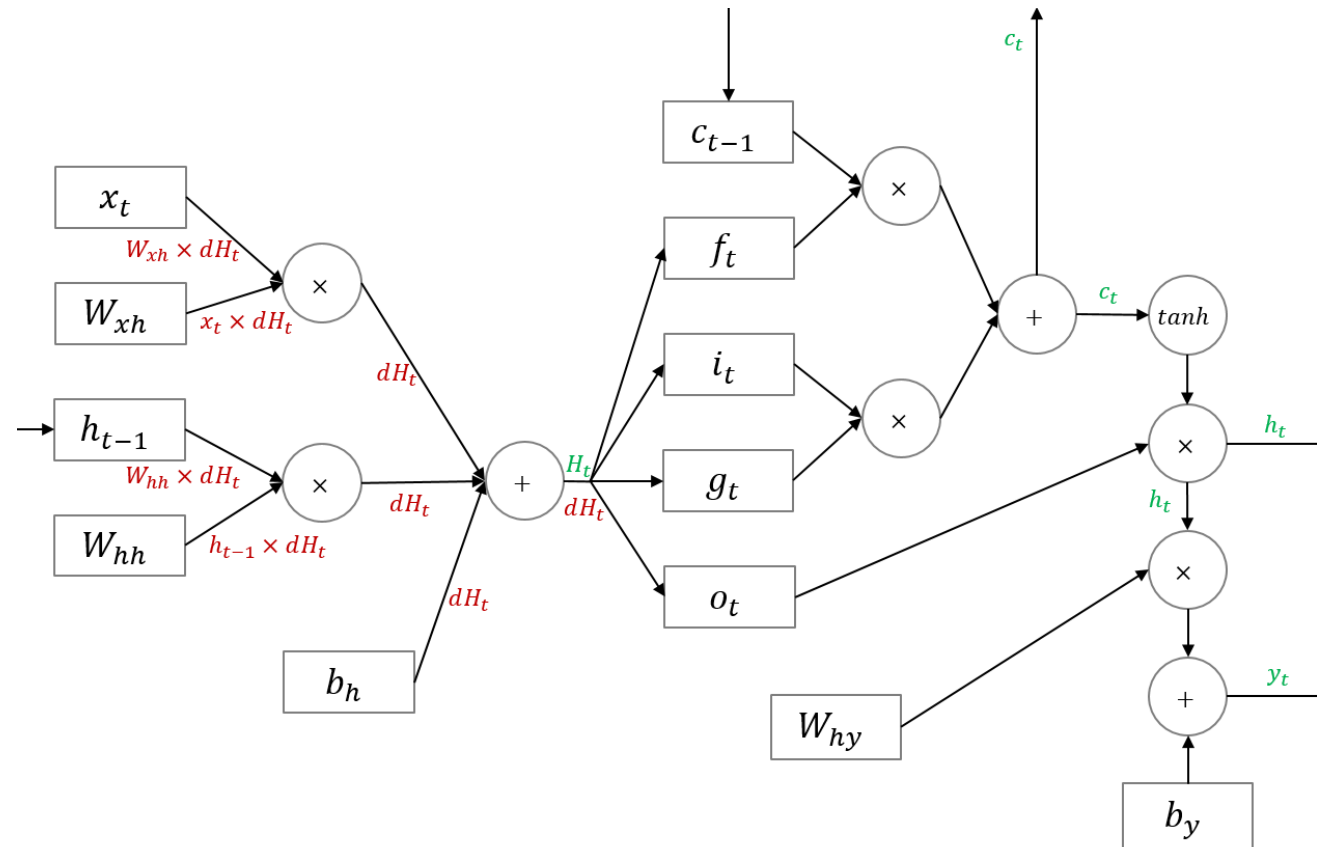
LSTM - LSTM Cell Backpropagation

- RNN 에서의 Backpropagation 과 유사



LSTM - LSTM Cell Backpropagation

- RNN 에서의 Backpropagation 과 유사 Details → <http://arunmallya.github.io/writeups/nn/lstm/index.html#/>



LSTM - Peephole connection

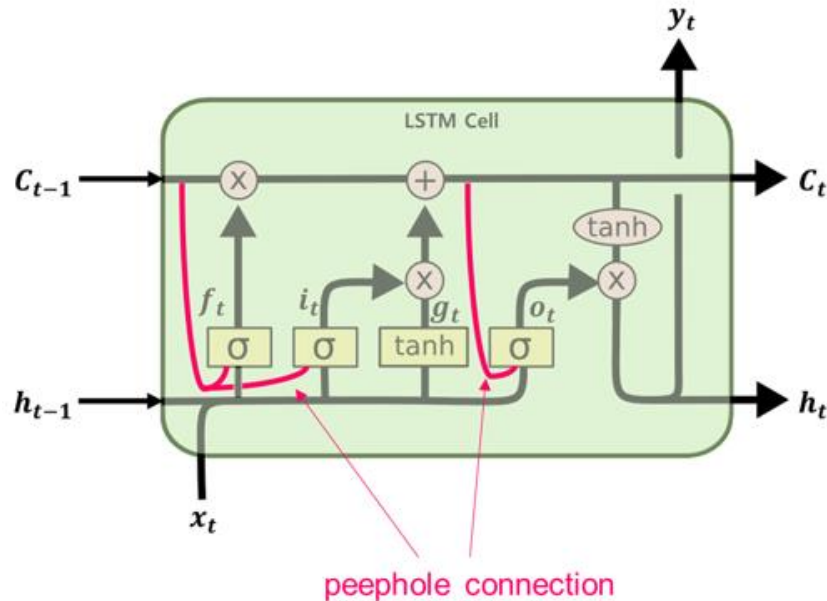
DNN

RNN

LSTM

GRU

- F. Gers and J. Schmidhuber (2000)
- 이전 Time step의 Cell state가 입력으로 추가되어 보다 많은 Context 인식



$$f_t = \sigma (\mathbf{W}_{cf}^T \cdot \mathbf{c}_{t-1} + \mathbf{W}_{xf}^T \cdot \mathbf{x}_t + \mathbf{W}_{hf}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_f)$$

$$i_t = \sigma (\mathbf{W}_{ci}^T \cdot \mathbf{c}_{t-1} + \mathbf{W}_{xi}^T \cdot \mathbf{x}_t + \mathbf{W}_{hi}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_i)$$

$$o_t = \sigma (\mathbf{W}_{co}^T \cdot \mathbf{c}_t + \mathbf{W}_{xo}^T \cdot \mathbf{x}_t + \mathbf{W}_{ho}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_o)$$

4

GRU

GATED RECURRENT UNIT

GRU - Gated Recurrent Unit

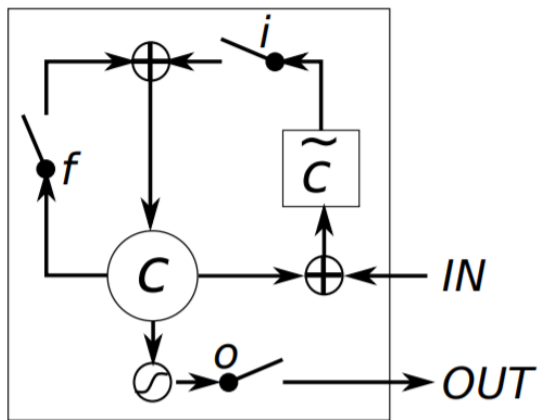
DNN

RNN

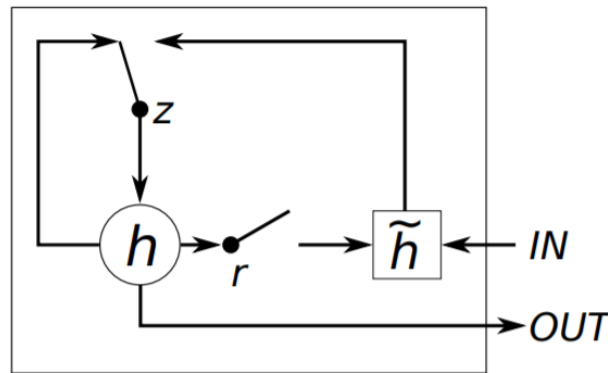
LSTM

GRU

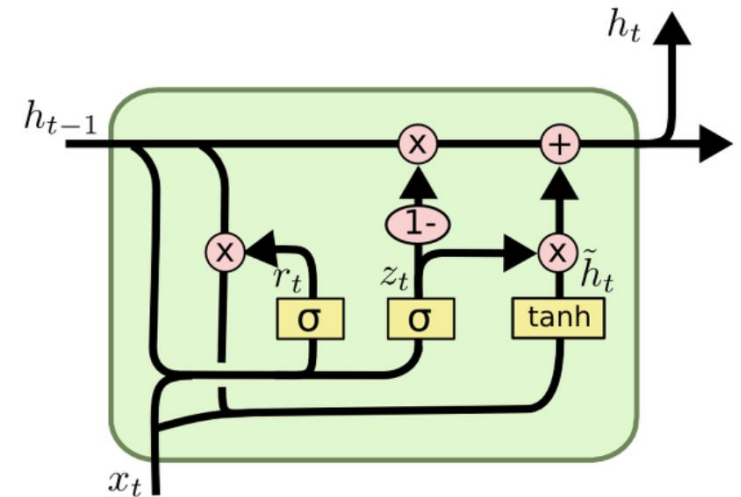
- [KyungHyun C. et al. \(2014\)](#)
- 기존 LSTM 에서 Cell state 와 Hidden state 가 하나로 통합되어 저장됨
- LSTM Cell의 Forget gate, input gate, output gate 대신 Update gate, reset gate 제시



(a) Long Short-Term Memory



(b) Gated Recurrent Unit



GRU - Gated Recurrent Unit

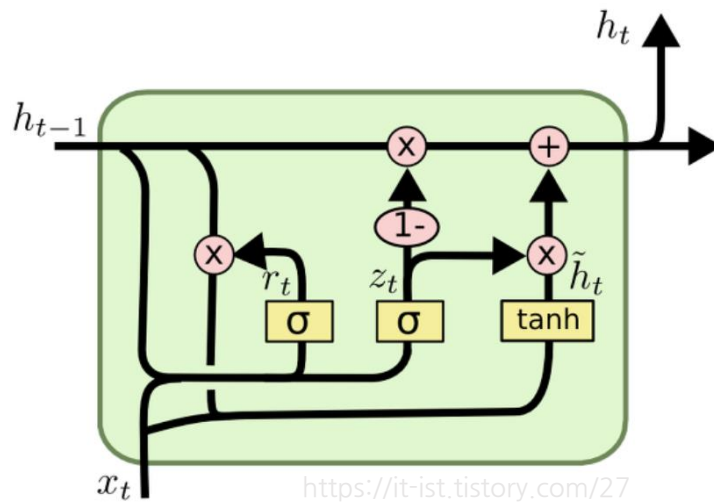
DNN

RNN

LSTM

GRU

- Update gate (z_t) : Take a linear sum between the existing state and the newly computed state
- Reset gate (r_t) : If it is reading the first symbol of an input sequence, allow it to forget
- GRU에는 LSTM의 Output gate mechanism이 존재하지 않고 r_t 가 Sequence의 시작만 구별



$$\begin{aligned}h_t^j &= (1 - z_t^j)h_{t-1}^j + z_t^j \tilde{h}_t^j \\z_t^j &= \sigma (W_z \mathbf{x}_t + U_z \mathbf{h}_{t-1})^j \\ \tilde{h}_t^j &= \tanh (W \mathbf{x}_t + U (\mathbf{r}_t \odot \mathbf{h}_{t-1}))^j \\ r_t^j &= \sigma (W_r \mathbf{x}_t + U_r \mathbf{h}_{t-1})^j\end{aligned}$$

GRU - Gated Recurrent Unit

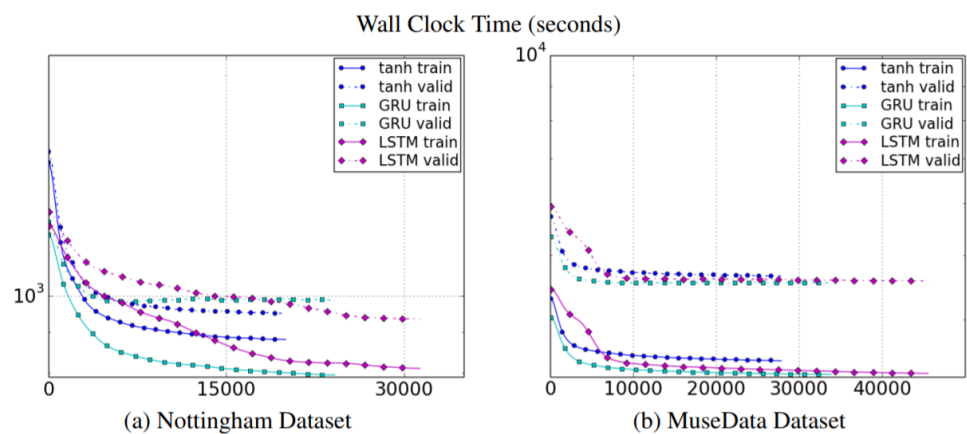
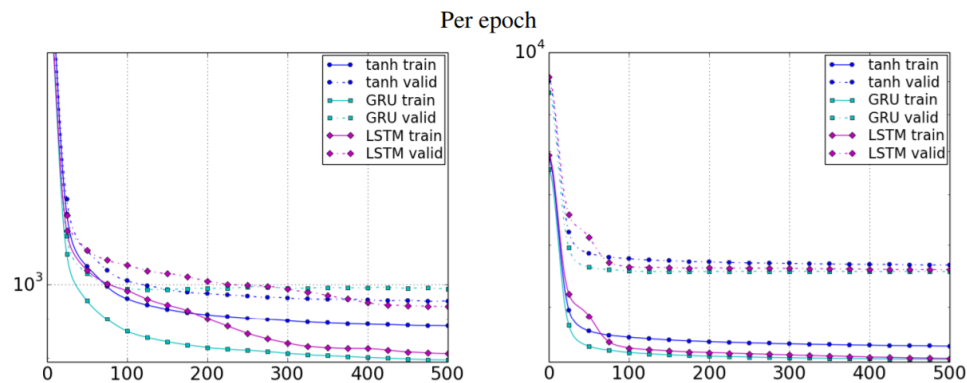
DNN

RNN

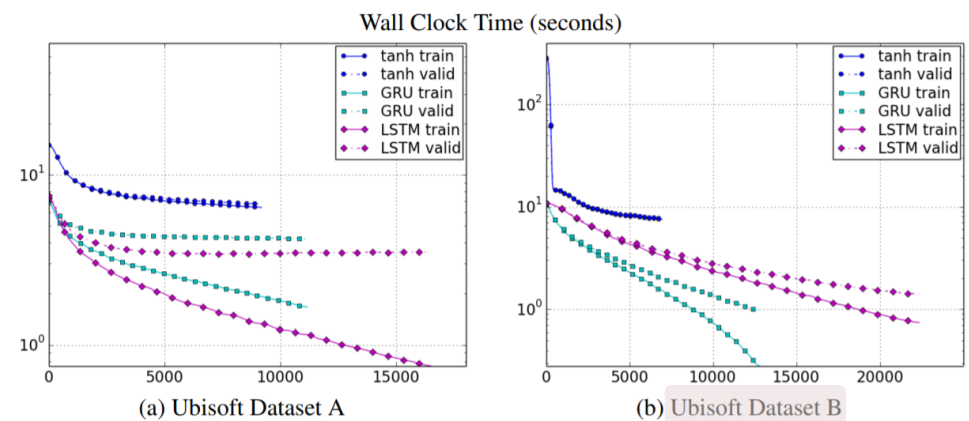
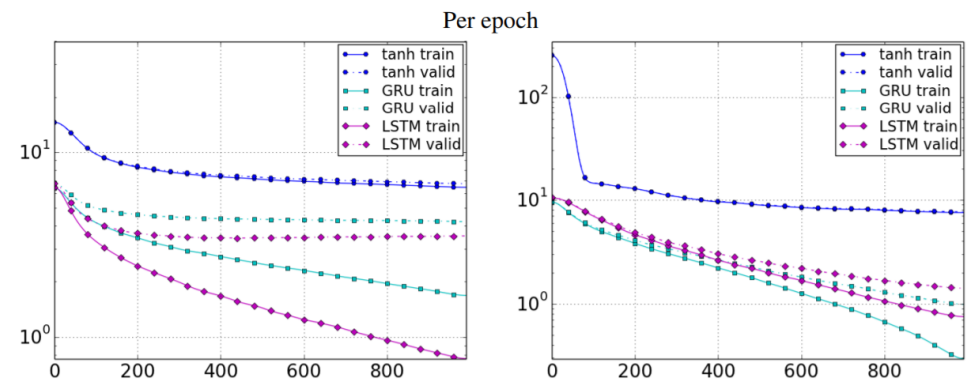
LSTM

GRU

- Traditional RNN 보다 나은 성능, LSTM과 비교해서는 Dataset의 종류에 영향을 받음
- 데이터 양이 적을 때 강세를 보임 (Ubisoft Dataset B)



Polyphonic music modeling



Speech signal modeling