Long Short-Term Memory (LSTM):

장단기메모리

https://youtu.be/Cl7ccWCAbtI

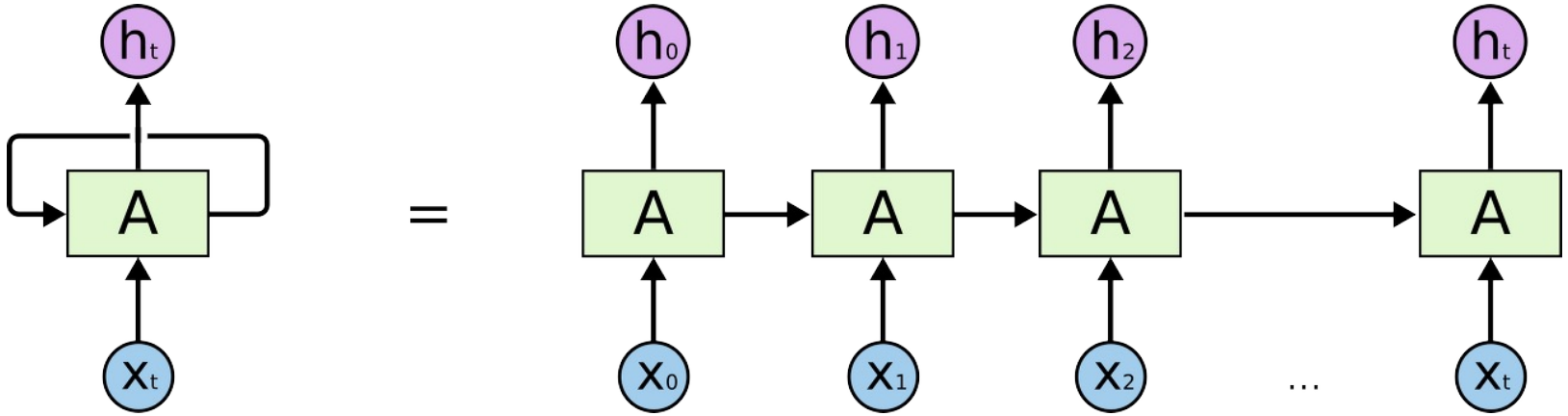
RNN이란?

RNN의 문제점

LSTM

RNN이란?

- **RNN (Recurrent Neural Network) : 순환신경망**
  - DNN, CNN, AE 같은 신경망은 은닉층에서 활성화 함수를 지난 후에 출력됨 → **피드 포워드 신경망**
  - 은닉층의 노드에서 활성화 함수를 통해 나온 결과값을 출력으로도 보내기도 하고 다음 은닉층 노드에 입력도 함
  - 출력값을 다음 은닉층 노드에 입력하므로 이전 단계에서 얻은 정보가 지속됨
    → 시계열 데이터를 다루기에 최적화된 신경망

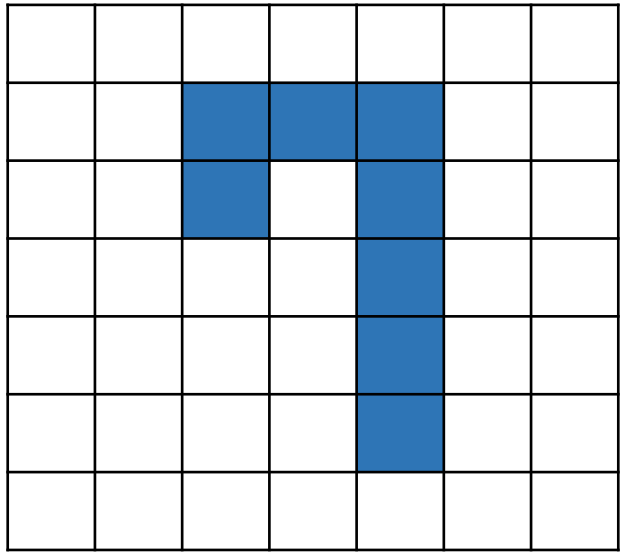


RNN의 구조

RNN이란?

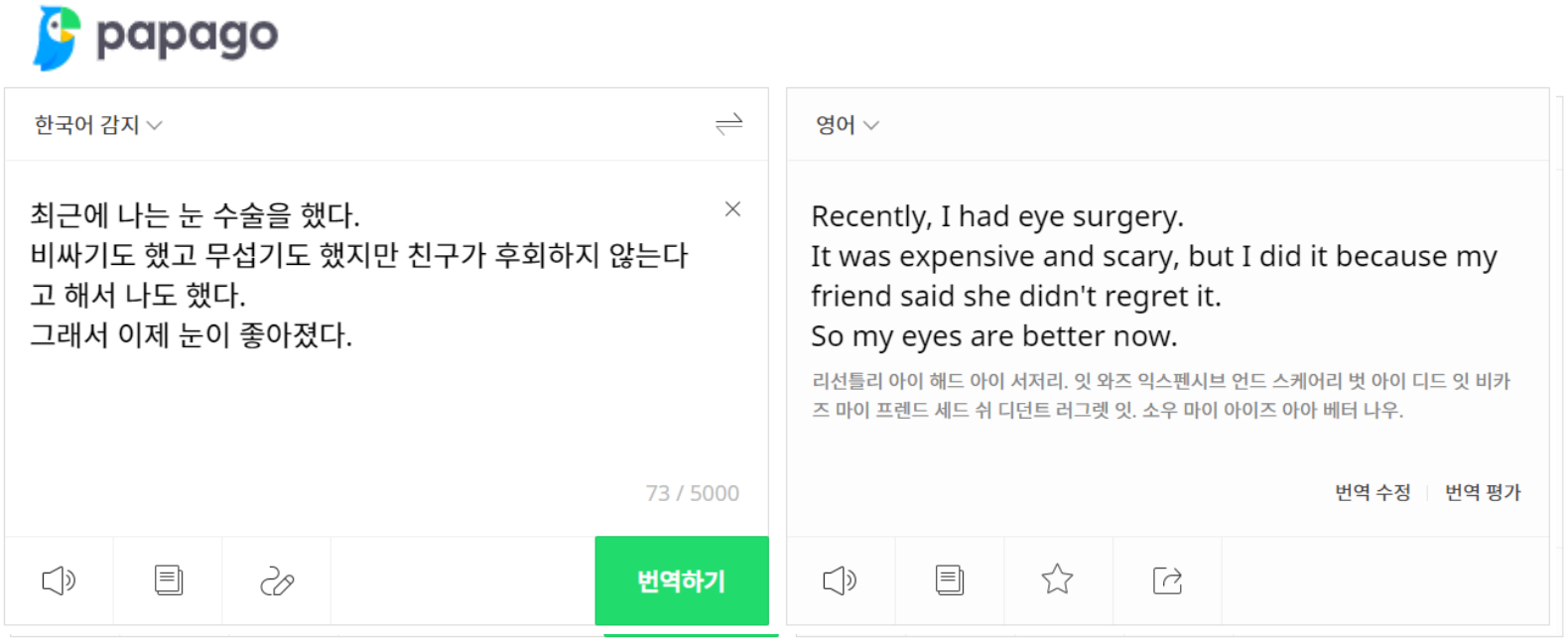
- **RNN (Recurrent Neural Network) : 순환신경망**
  - DNN, CNN, AE 같은 신경망은 은닉층에서 활성화 함수를 지난 후에 출력됨 → **피드 포워드 신경망**
  - 은닉층의 노드에서 활성화 함수를 통해 나온 결과값을 출력으로도 보내기도 하고 다음 은닉층 노드에 입력도 함
  - 출력값을 다음 은닉층 노드에 입력하므로 이전 단계에서 얻은 정보가 지속됨

    → 시계열 데이터를 다루기에 최적화된 신경망



RNN이란?

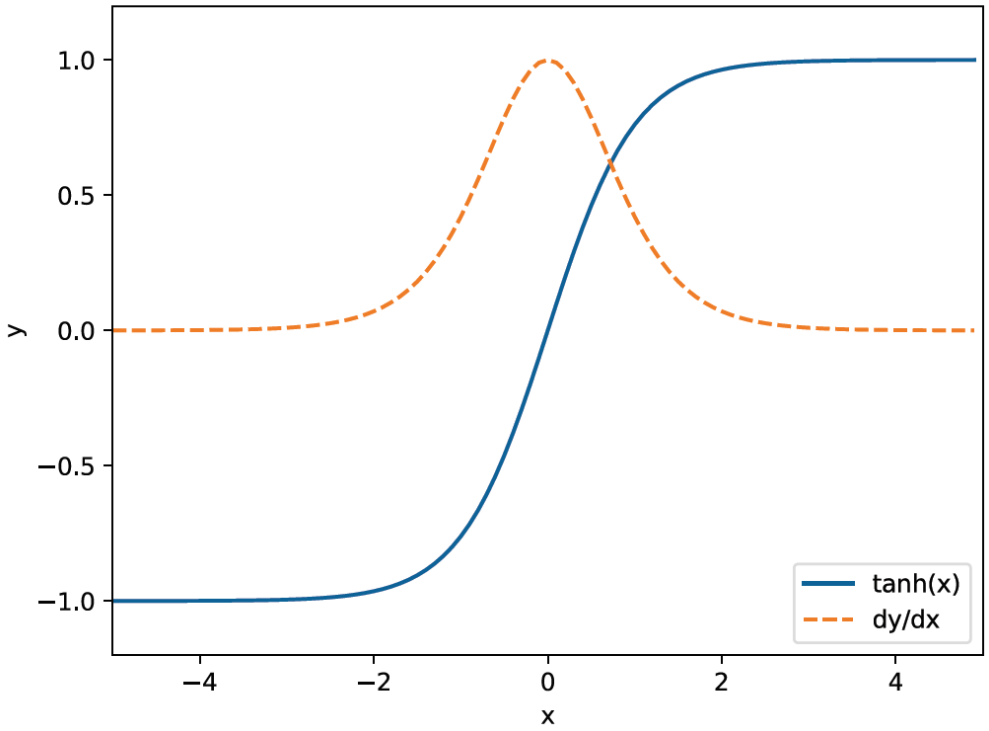
- **RNN의 문제점 : 장기의존성**
  - 은닉층의 과거의 정보가 마지막까지 전달되지 못하는 현상을 의미
  - 무언가를 얻기 위해서 최근의 정보만이 필요할 때도 있지만, 과거의 정보가 필요한 경우가 있을 수 있음
  - 필요한 정보를 얻기 위한 시간 격차 ⬆ RNN의 성능 ⬇



RNN이란?

- **장기의존성 문제의 원인**
  - Tanh 활성화 함수의 기울기는 0과 1사이

그림 6-6 $y = \tanh(x)$의 그래프(점선은 미분)



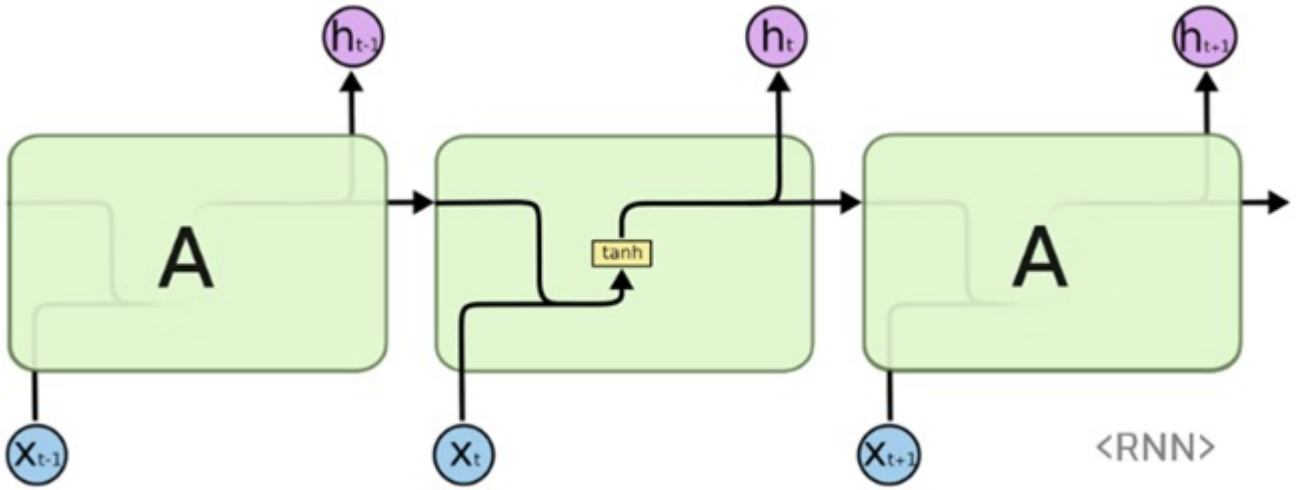
RNN이란?

- **RNN의 장기의존성 문제**
  - Tanh 함수의 기울기가 0과 1 사이이므로 결국 0으로 수렴 → 기울기 소실 → 성능 하락

$$h_{t-2} = tanh(W[h_{t-3}, x_{t-2}])$$

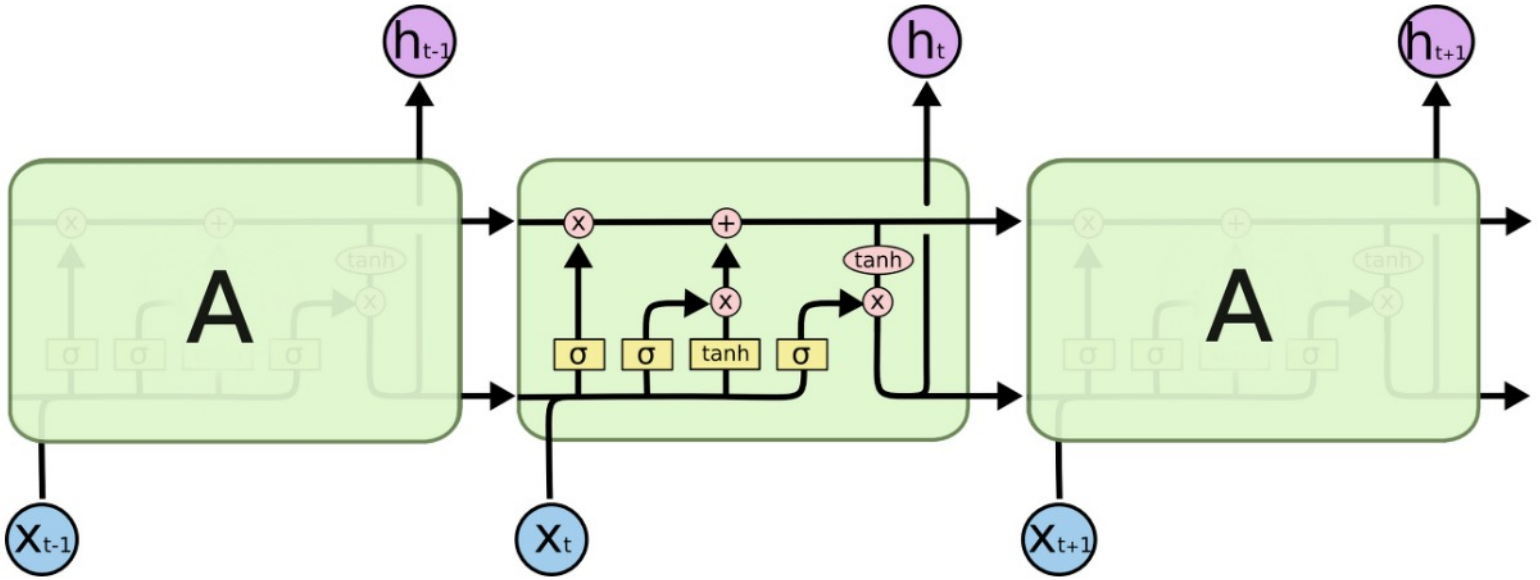
$$h_{t-1} = tanh(W[h_{t-2}, x_{t-1}])$$

$$h_t = tanh(W[h_{t-1}, x_t]) \rightarrow h_t = tanh(W[tanh(..tanh(..h_{t-3})), x_t])$$



LSTM이란?

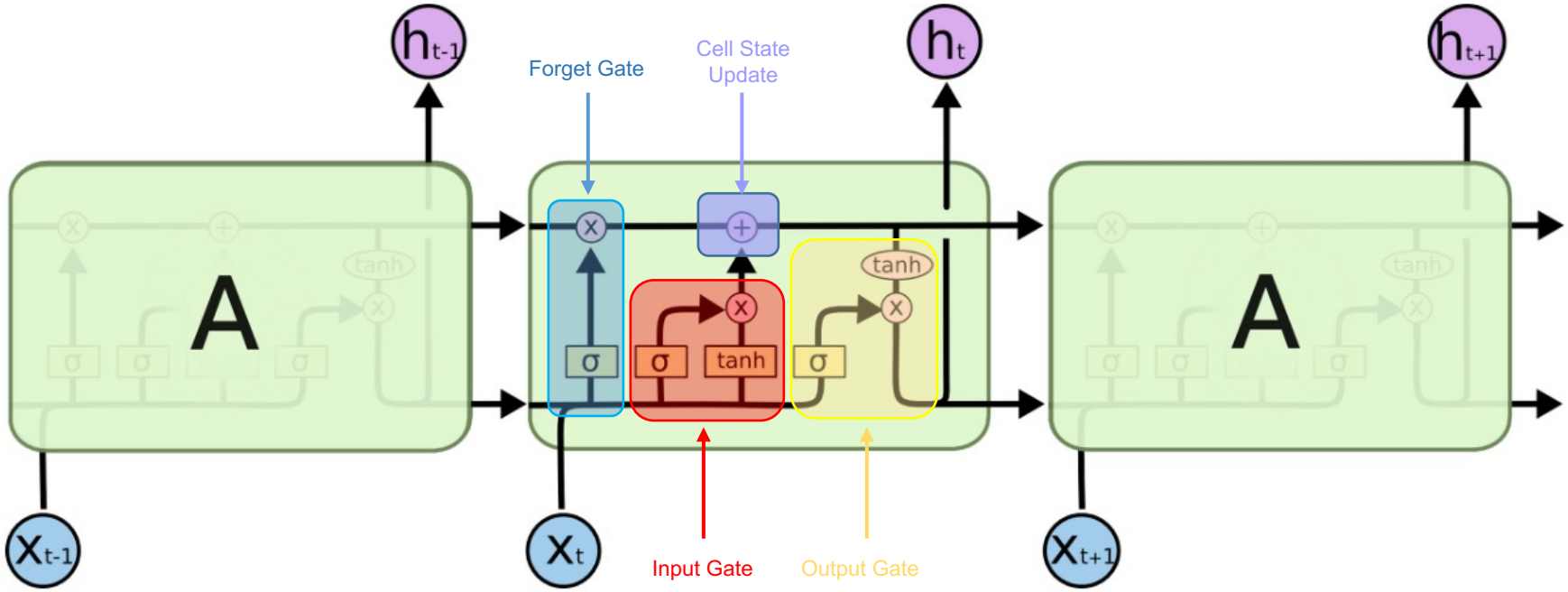
- **Long Short-Term Memory (LSTM): 장단기메모리**
  - 3개의 Gate를 추가함으로써 기울기 소실 문제를 해결하여 RNN의 문제점을 극복한 모델
  - 요즘은 RNN을 거의 사용 X



LSTM이란?

- **Long Short-Term Memory (LSTM): 장단기메모리**

  1. Forget Gate : 과거 정보를 얼마나 유지할 것인가?
  2. Input Gate : 새로 입력된 정보를 얼마나 장기 상태에 활용할 것인가?
  3. Cell State Update : Forget Gate와 Input Gate에서 구한 값들을 통해 C값 갱신
  4. Output Gate : 출력 정보를 얼마나 넘겨줄 것인가?



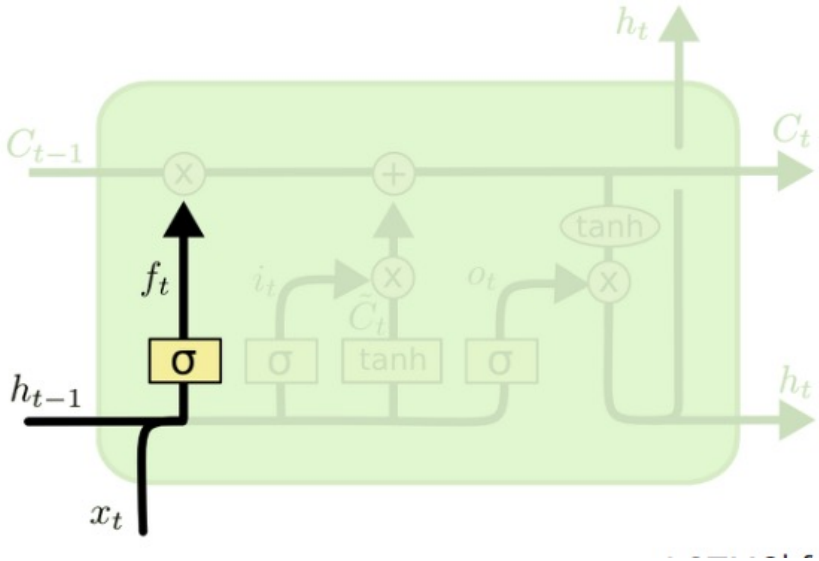
LSTM이란?

1. Forget Gate : 과거 정보를 얼마나 유지할 것인가?

1) h_{t-1}과 x_t에 각각 가중치를 곱한 후 편향 더하기

2) 시그모이드 함수를 적용하여 0과 1 사이의 값을 C_{t-1}로 보내줌

→ 두 과정을 통해 과거의 불필요한 데이터 **제거**



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

- 0 : 과거의 모든 정보 **제거**
- 1 : 과거의 모든 정보 **보존**

h_{t-1} : 과거의 정보

x_t : 현시점의 정보

LSTM이란?

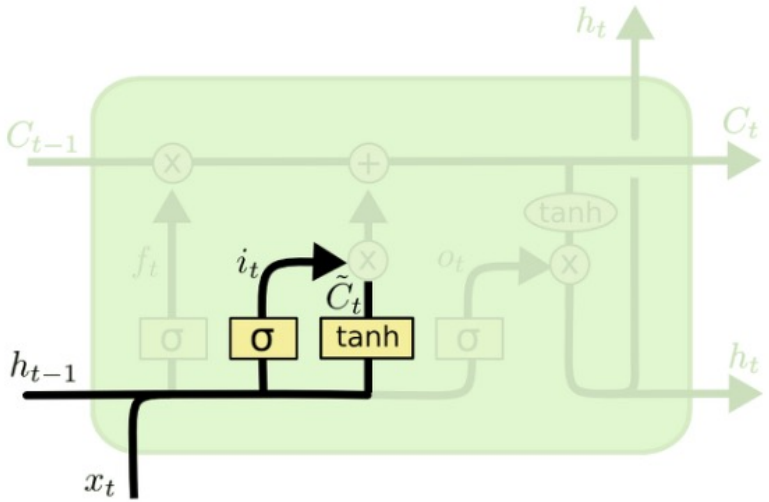
2. Input Gate : 새로 입력된 정보를 얼마나 장기 상태에 활용할 것인가?

1) h_{t-1}과 x_t에 각각 가중치를 곱한 후 편향 더하기

2) 시그모이드 함수를 적용하여 0과 1 사이의 값으로 변환 (i_t)

3) tanh 함수를 적용하여 -1과 1 사이의 값으로 변환 ($\tilde{C}_t$)

→ 두 값을 통해 현시점 정보의 기억할 양을 정함



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

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

h_{t-1} : 과거의 정보

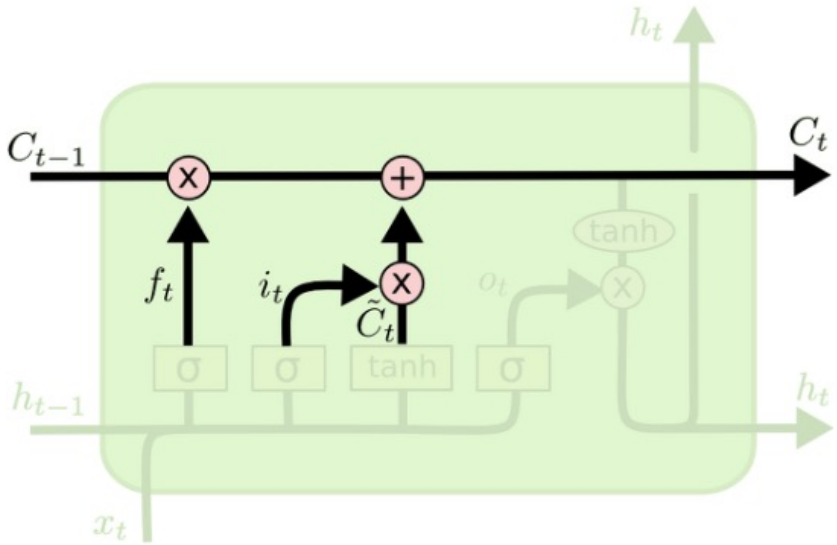
x_t : 현시점의 정보

LSTM이란?

3. Cell State Update : Forget Gate와 Input Gate에서 구한 값들을 통해 C값 갱신

1) 이전 Cell State와 앞서 Forget Gate에서 구한 값 (f_t)를 곱해줌 → 과거의 데이터에서 불필요한 정보 **제거**

2) 해당 값을 Input Gate에서 구한 값 i_t와 $\tilde{C}_t$를 곱한 값과 더해줌 → 현시점의 데이터에서 중요한 정보 **삽입**



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

h_{t-1} : 과거의 정보

x_t : 현시점의 정보

LSTM이란?

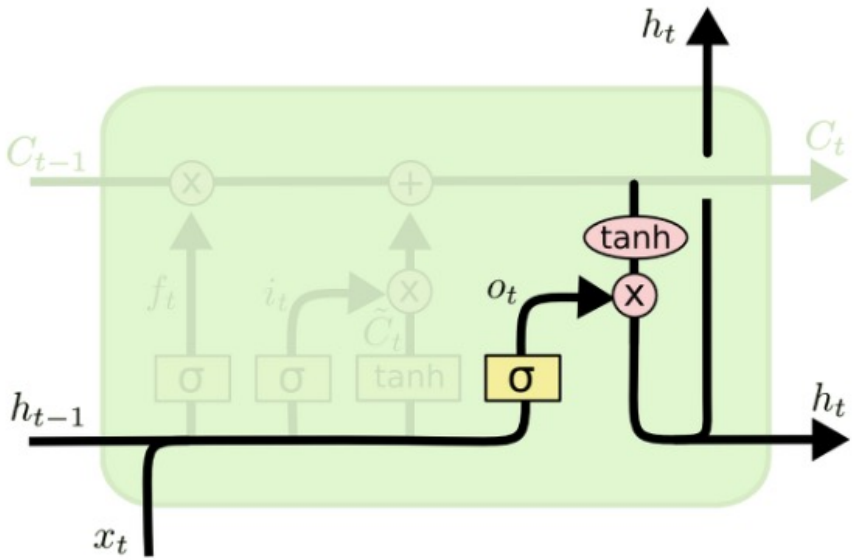
4. Output Gate : 출력 정보를 얼마나 넘겨줄 것인가?

1) h_{t-1}과 x_t에 각각 가중치를 곱한 후 편향 더하기

2) 시그모이드 함수를 적용하여 0과 1 사이의 값으로 변환 (o_t)

3) 갱신된 C_t와 o_t를 곱해줌 (h_t)

→ h_t를 통해 출력 정보를 얼마나 넘겨줄지 정함



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

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

h_{t-1} : 과거의 정보

x_t : 현시점의 정보

Q & A