Seoul
Software
ACademy

실무요건 기반 핀테크 데이터분석가 과정

- 딥러닝 RNN -

핀인사이트
데이터분석가 김현진
jinny@fins.ai

RNN

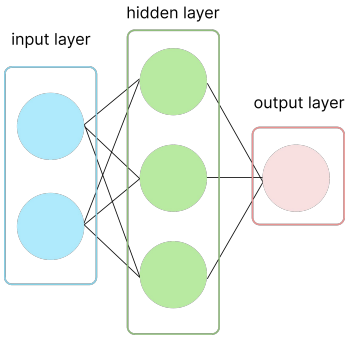
RNN(Recurrent Neural Network)

✔ RNN(순환신경망)

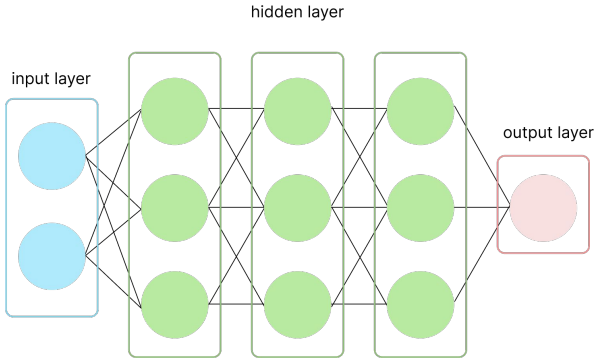
- 가장기본적인 인공신경망 시퀀스 모델로, 입력과 출력을 시퀀스 단위로 처리하는 모델

■ 순차데이터(sequential data)
: '순서'를 가진 데이터로 순서가 변경될 경우 데이터의 특성을 잃어버리는 데이터를 의미함
예) 문장, 주가, 날씨, DNA 유전정보 등의 시계열 데이터(time series data)

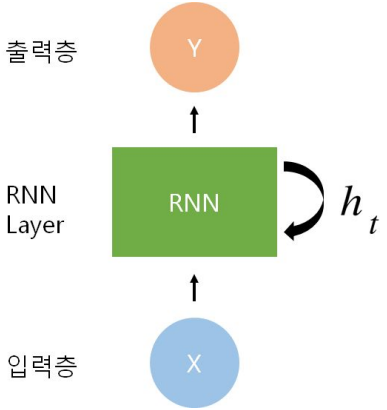
Neural Network와 RNN의 구조적 차이



Neural Network



Deep Neural Network



RNN

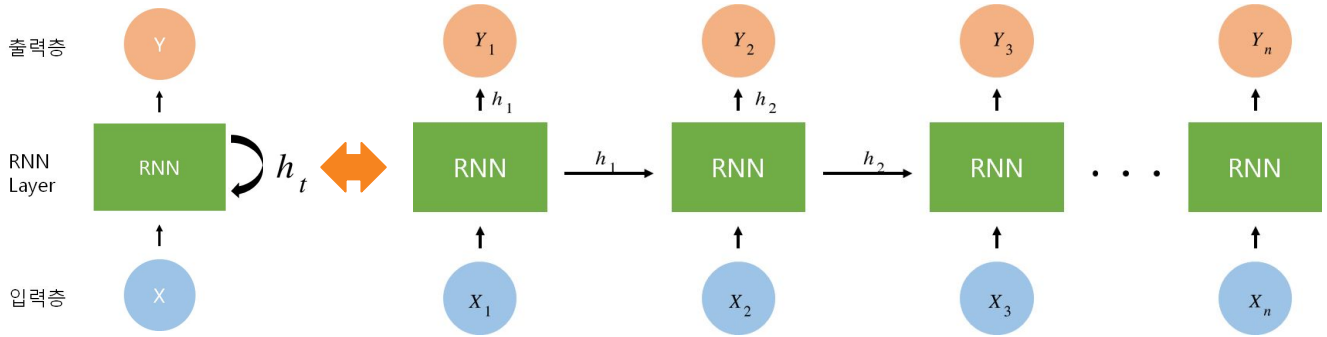
Neural Network의 문제점

- 뉴럴네트워크(Neural Network)의 경우 각 레이어에 사건이나 단어에 대한 정보를 담고있음
- 하지만, 새로운 단어와 사건에 대해 예측하는 경우 이전 단어와 사건을 기반하기 때문에 새로 등장한 단어를 예측하지 못한다는 단점이 있음
- 이 문제를 해결하기 위해 등장한 것이 RNN모델
- 즉, RNN의 경우 사건/문맥의 앞에 주어진 정보의 흐름을 활용하여 정답을 예측할 수 있는 모델

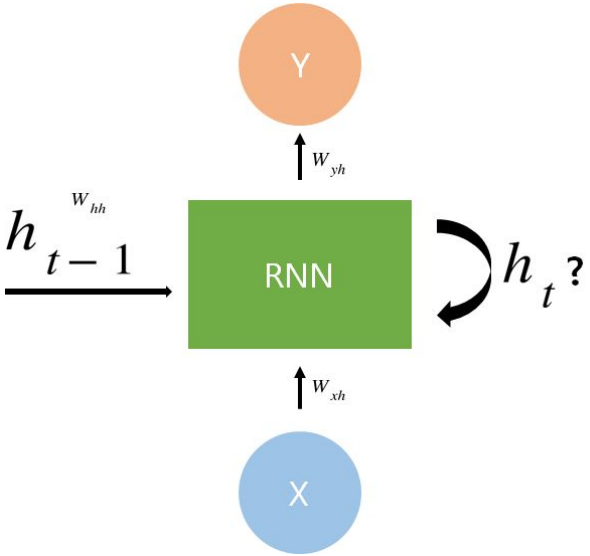
RNN 구조적 특징

✔ RNN 기본구조

- RNN Layer = 순환층 = cell
- RNN Layer의 출력 결과 = hidden state = hidden vector
- hidden state로 인한 정보의 지속성



hidden state 계산



$$h_t = f_w(h_{t-1}, X_t) = = \tanh(W_{hh}h_{t-1} + W_{xh}X_t)$$

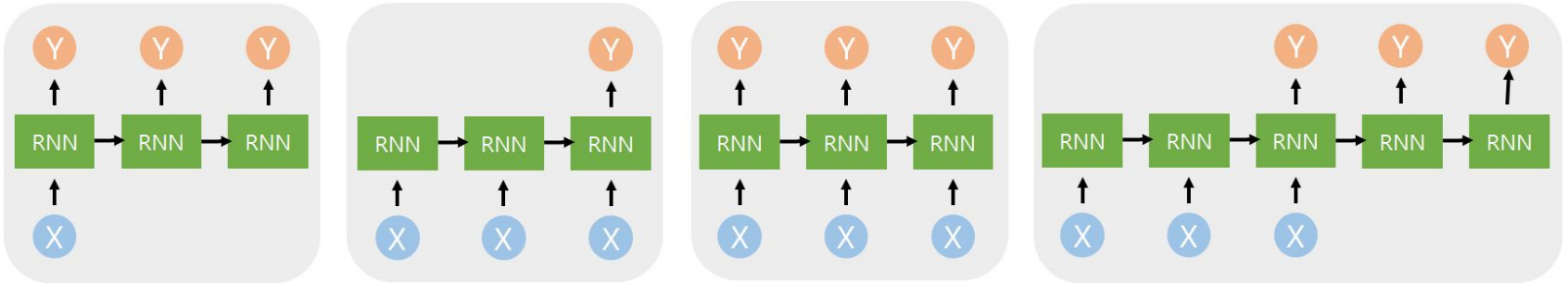
h_t	- 현재 RNN셀에서 계산한 히든 스테이트 - 다음 RNN셀의 입력으로 전달됨
h_{t-1}	- 이전 RNN셀에서 계산한 히든 스테이트 - 현재 RNN셀의 입력으로 전달됨 - 첫번째 RNN셀은 이전 히든 스테이트가 없음(h_{t-1} =0)
W_{hh} , W_{xh} , W_{yh}	- 학습 대상이 되는 가중치

위의 식을 사용해 hidden state를 계산 했다면,
아래 식을 사용해 예측값을 구할 수 있음

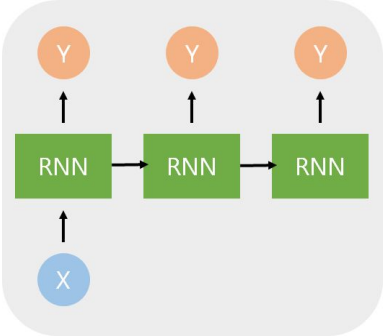
$$Y_t = W_{hy}h_t$$

RNN 종류

RNN은 아래 그림과 같이 4가지 응용모형이 있음



RNN 종류



✔ One to Many

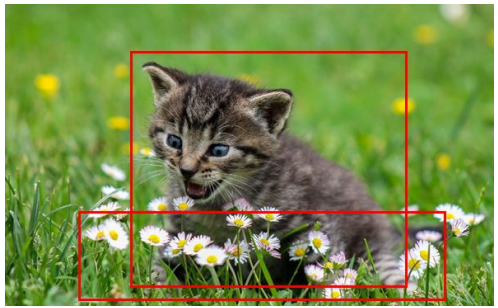
: 하나의 입력을 받아 RNN 층을 통과하면 여러개의 출력값을 내보내는 모형

- 출력갯수는 정해져 있지 않고 사용자가 결정함

■ 이미지 캡셔닝

: 컴퓨터가 이미지 데이터를 입력받은 후 이미지에 적합한 설명을 자동으로 붙여주는 방법

- 입력값으로 이미지 데이터를 넣으면 출력값으로 이미지를 설명하는 문장이 나옴(여러개 가능)



RNN

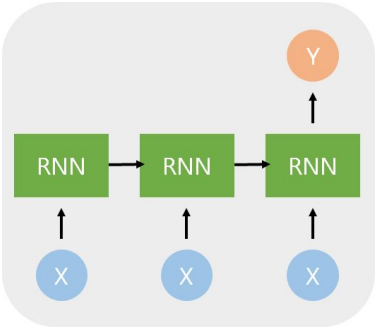
A cat sitting in the middle.

Flowers bloom in front of the cat.

RNN 종류

✔ Many to One

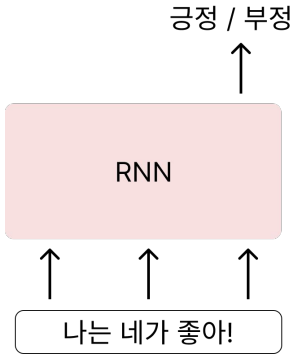
: 여러개 입력을 받아 RNN층을 통과하여 하나의 출력을 내보내는 모형



■ 감정분석

: 어떤 문장이나 문서에 대한 주관적 극성을 분류하는 방법

- 문서(문장)을 토큰화하고, 토큰화된 단어는 여러개의 입력이 되어 RNN층을 통과하게 되면 긍정 혹은 부정의 확률값이 출력됨. 확률값을 보고 긍정,부정 판단 가능

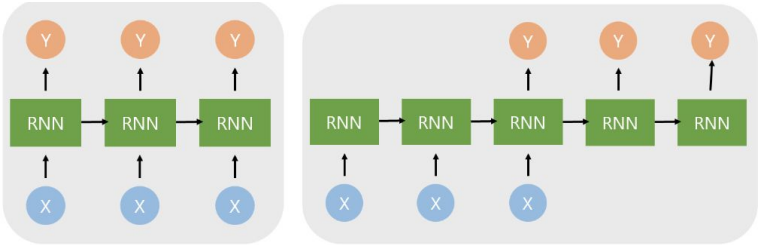


RNN 종류

✔ Many to Many

: 여러개의 입력값을 받아 RNN층을 통과하여 여러개의 출력값을 내모내는 모형

- 아래 그림과 같이 입력과 출력의 갯수가 같은경우와, 같지않은 경우 두가지 모형이 존재함
- 입력과 출력의 갯수가 같은 경우를 우는 'Seq2Seq'라 부름



■ 개체명인식

: 단어가 입력되면, 입력된 단어가 어떤 유형(사람, 장소, 조직, 시간 등)에 속하는지 인식하는 방법

- 문서(문장)을 토큰화하고, 토큰화된 단어는 여러개의 입력이 되어 RNN층을 통과하게 되면 긍정 혹은 부정의 확률값이 출력됨. 확률값을 보고 긍정,부정 판단 가능

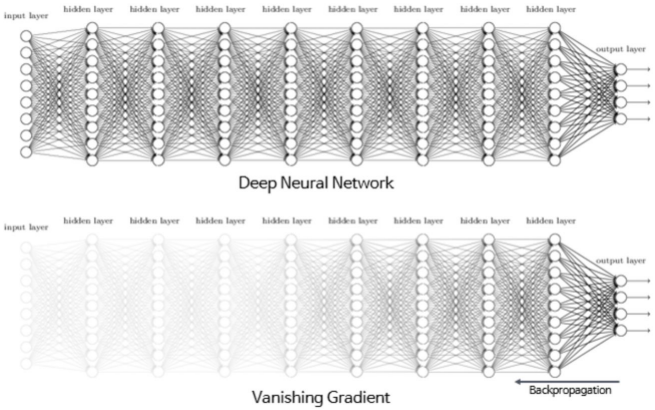
■ 기계번역

RNN 한계점

✔ 장기의존성 문제

- 예측하는 문장의 길이가 길어지는 경우 hidden state를 통해 넘겨받게 되는 정보의 비중이 점차 줄어듬. 때문에, 먼저 입력된 단어의 반영도는 점차 줄어드는 문제가 발생

✔ Gradient Vanishing / Gradient Exploding



* back propagation
: 데이터가 입력층에서 출력층으로 가는 것과 반대로 출력층에서 입력층 방향으로 가중치를 업데이트 하는 방법으로, Layer가 많아질수록 가중치를 계속 곱하다보면 값이 0에 수렴하여 기울기가 소실됨

실습1

SimpleRNN 실습

LSTM

LSTM(Long Short Term Memory)

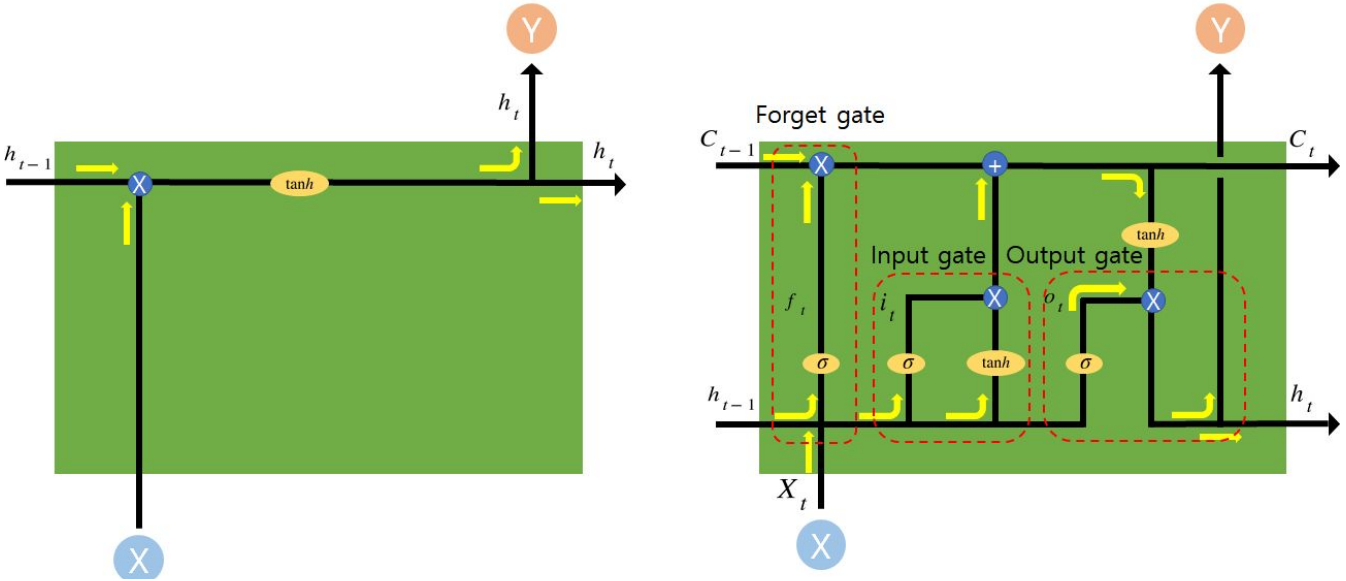
✔ LSTM(장단기 메모리)

- 1997년 S.Hochreiter와 J.Schmidhuber가 제안
- RNN의 장기의존성문제와 Gradient 문제를 해결하기 위해 고안된 모형
- ‘Gate’를 추가하여 기억할것과 잊을것을 선택하여 중요한 정보만 기억하도록 함
- 기존 RNN 모형에비해 학습속도가 빨라짐

참고 : https://colah.github.io/posts/2015-08-Understanding-LSTMs/

RNN과 LSTM의 구조적 차이

- 기존 RNN에는 단기상태(short-term state)를 의미하는 h_t 만 존재했는데, LSTM에는 Cell State가 추가됨
- LSTM은 장기상태(Long-term state)를 의미하는 C_t벡터가 추가됨

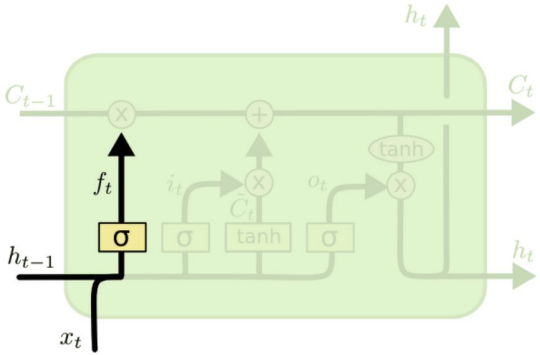


LSTM 구조

✔ Forget Gate

: 전 단계에서 받은 정보를 얼마나 잊어버릴지에 대한 값을 결정함

- 입력값으로 h_{t-1}와 X_t를 받아 계산한 후 시그모이드(σ)를 통해 0 ~ 1 사이 값으로 변환하면 이전 정보를 얼마나 잊어버릴지에 대한 계산



$$f_t = \sigma\left(W_{xf}^T \cdot X_t + W_{hf}^T \cdot h_{t-1} + b_f \right)$$

f_t	-	정보를 잊을 확률값
W^T_{xf}	-	forget gate 입력 가중치
X_t	-	입력
W^T_{hf}	-	forget gate 히든 스테이트 가중치
h_{t-1}	-	이전 히든 스테이트
b_f	-	편향

LSTM 구조

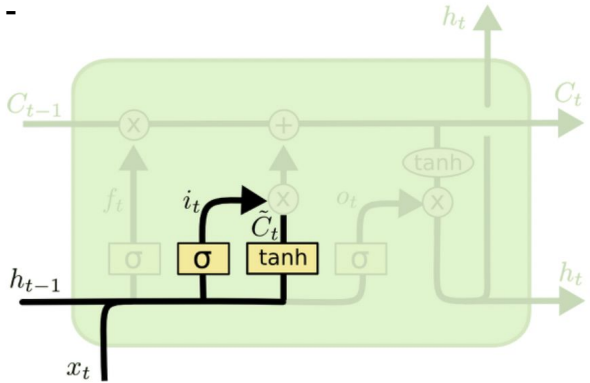
✔ 현재 Cell의 정보 $\tilde{C}$

- : 현재 cell의 정보를 얼마만큼 반영할지에 대한값을 결정함
- 입력값으로는 h_{t-1}와 X_t를 받아 계산하고 하이퍼블릭 탄젠트를 취해 현재 Cell의 정보를 얼마나 반영할지를 계산

✔ Input Gate

: 현재 cell의 정보인 $\tilde{C}$ 를 얼마나 반영할것인가에 대한 값을 계산

- 입력값으로 X_t와 h_{t-1}을 입력받아 Input Gate를 생성한 후 시그모이드 함수를 통해 Forget Gate와 마찬가지로 0과 1사이의 값으로 변환
-



$$i_t = \sigma\left(W_{xi}^T \cdot X_t + W_{hi}^T \cdot h_{t-1} + b_i \right)$$

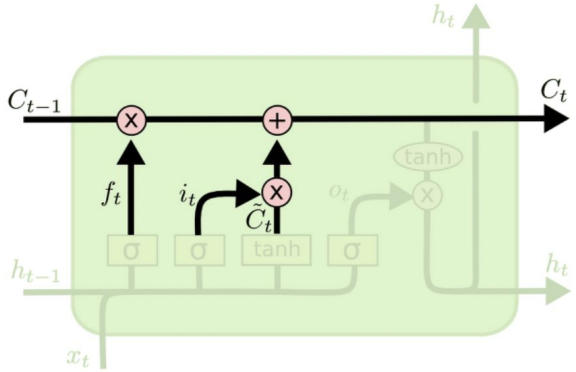
$$\tilde{C} = \tanh\left(W_{xg}^T \cdot X_t + W_{hg}^T \cdot h_{t-1} + b_g \right)$$

LSTM 구조

✔ Cell State(C_t)

: C_t의 값은 Forget Gate와 Input Gate에서 전 단계의 정보와 현재의 정보를 얼마나 잊어버리고 기억할지에 대한 합

- forget gate와 input gate 값이 사전에 먼저 계산되어야함
- C_t값은 바로 출력되어 다음 cell로 값이 넘어감



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

이전 단계의 Cell을 얼마나 잊어버릴 것인가

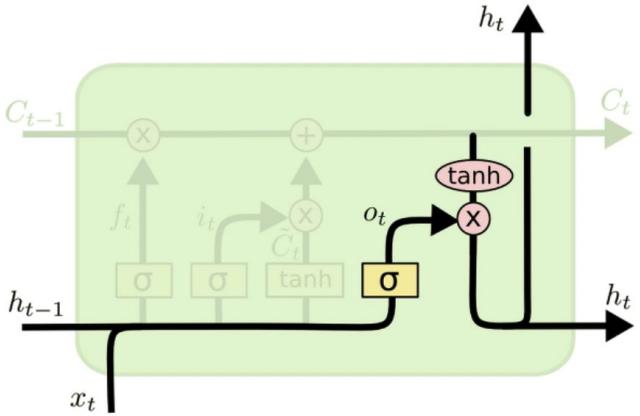
현재의 Cell을 얼마나 기억할 것인가

LSTM 구조

✔ Output Gate

: 다음 hidden state로 얼마만큼의 hidden vector(h_t)가 넘어가게 될지에 대한 값 계산

- 입력값으로 X_t와 h_{t-1}을 입력받아 output Gate를 생성한 후 시그모이드 함수를 통해 forget gate와 마찬가지로 0과 1사이의 값으로 변환



$$o_t = \sigma\left(W_{xo}^T \cdot X_t + W_{ho}^T \cdot h_{t-1} + b_o \right)$$

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

실습2

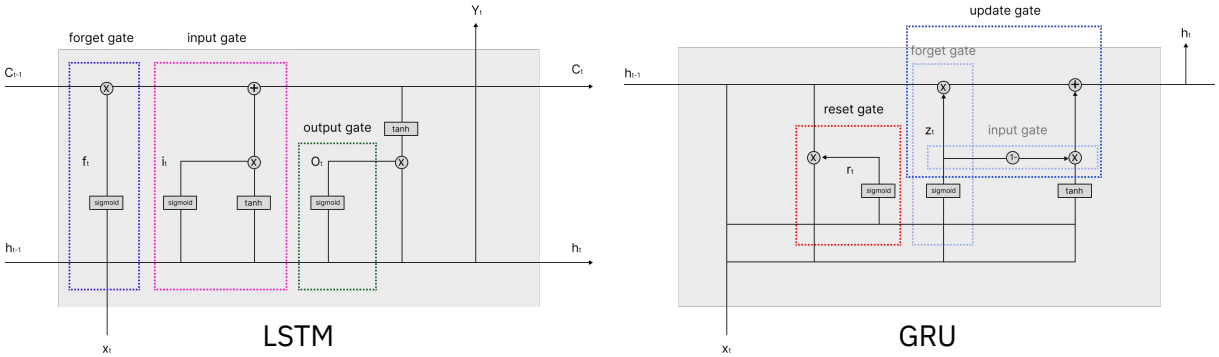
LSTM 실습

GRU

GRU(Gated Recurrent Unit)

✔ GRU(게이트 순환 유닛)

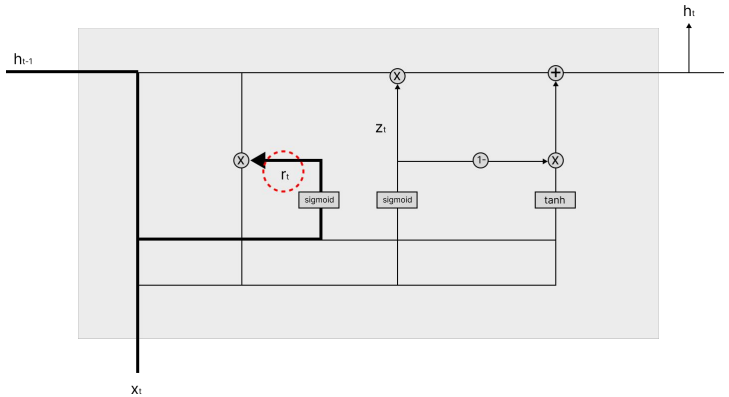
- 2014년 K.Cho(조경현 교수)가 제안한 모델로 LSTM을 간소화한 모델
  논문 “Learning Phrase Representations using RNN Encoder-Decoder for statistical Machine Translation”
  (LSTM gate 3개 -> GRU gate 2개)
- LSTM 모델보다 파라미터가 적어 연산비용이 적게들고, 학습속도가 빠름
- Forget Gate와 Input Gate를 합쳐 Update Gate를 만들고, Reset Gate를 추가함



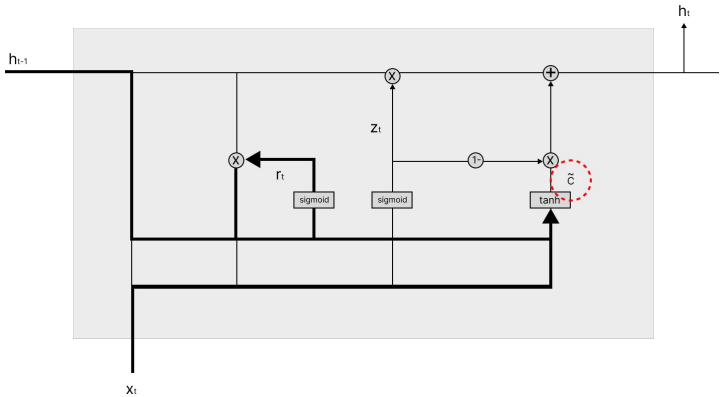
GRU 구조

✔ Reset Gate

: 이전 정보에서 얼마만큼을 선택하여 내보낼지에 대한 계산



$$r_t = \text{sigmoid}(w_r h_{t-1} + w_h h_{t-1} + b_r)$$

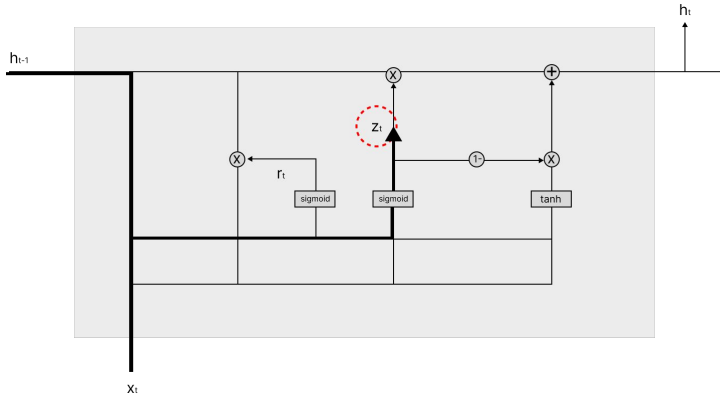


$$\tilde{c} = \tanh(w_x x_t + w_h(r_t h_{t-1}) + b_c$$

GRU 구조

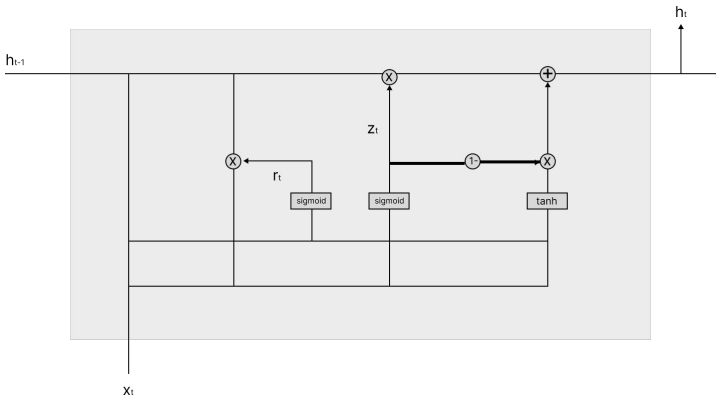
✔ Update Gate

: 과거정보를 얼마만큼 반영하고, 현재 정보를 얼마만큼 반영할것인가에 대한 계산



$z_t = \text{sigmoid}(w_x x_t + w_h h_{t-1} + b_z)$

⇒ forget gate



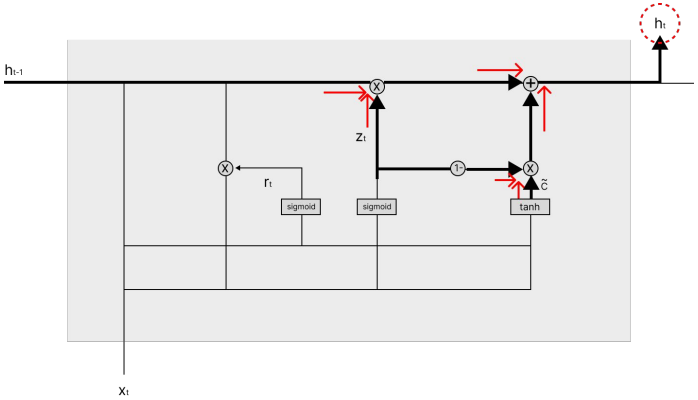
input gate = $1 - z_t$

GRU 구조

✔ hidden state

: 현재 셀의 값과 update gate의 결과를 결합하여 현재 시점의 hidden state 계산

- 이 값이 최종적으로 output으로 나가는 hidden state 값임



$$h_t = z_t h_{t-1} + (1 - z_t)\tilde{c}$$

실습3

GRU 실습