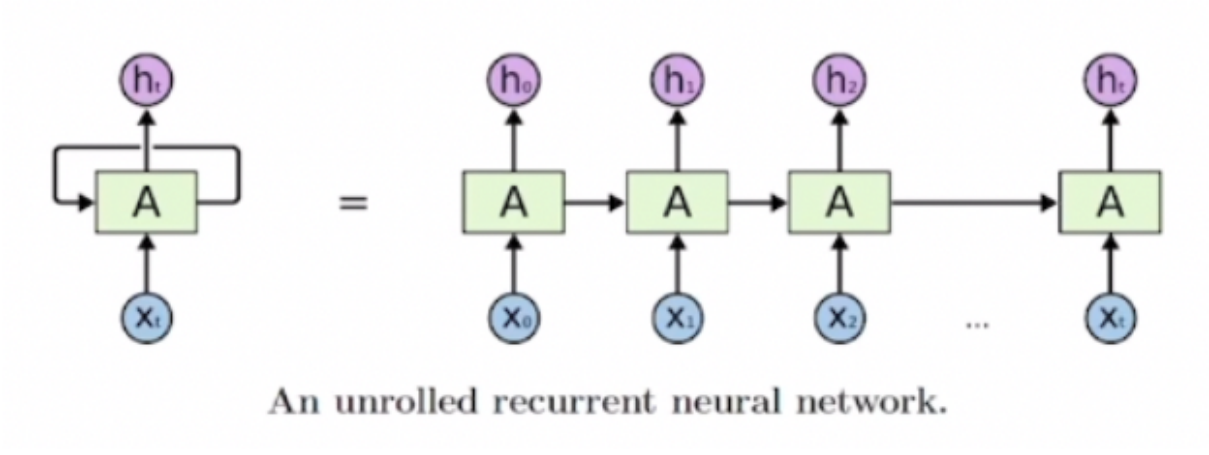
2. 자연어 처리와 딥러닝

RNN

RNN 구조와 계산 방법

Basic structure

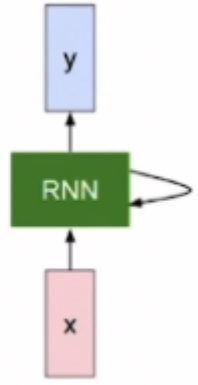


An unrolled recurrent neural network.

- 현재 타임스텝에 대해 **이전 스텝까지의 정보를 기반으로 예측값을 산출**하는 구조의 딥러닝 모델
- 매 타임스텝마다 **동일한 파라미터**를 가진 모듈을 사용하므로, 재귀적 호출의 특성을 보여줌

How to calculate the hidden state of RNNs

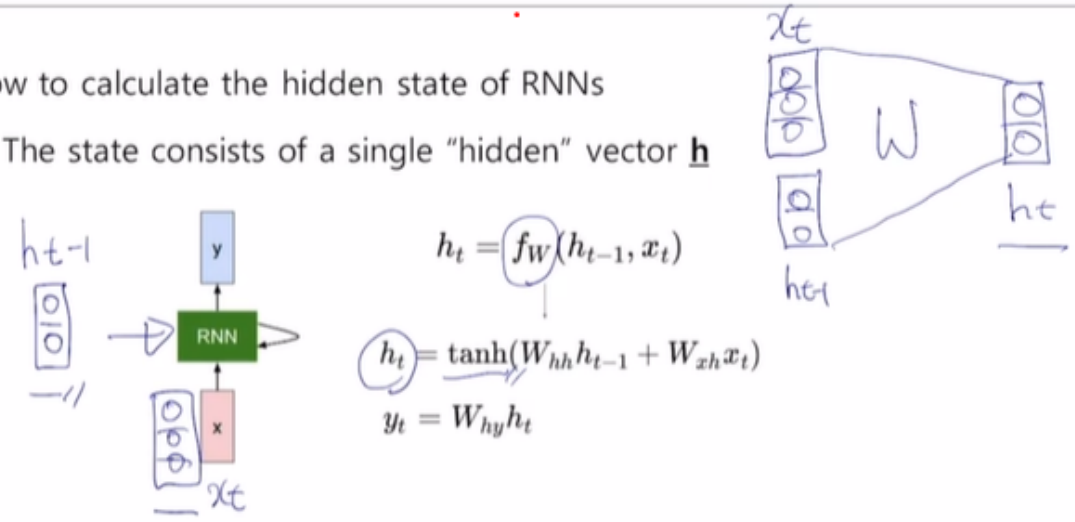
$$h_t = f_W(h_{t-1}, x_t)$$

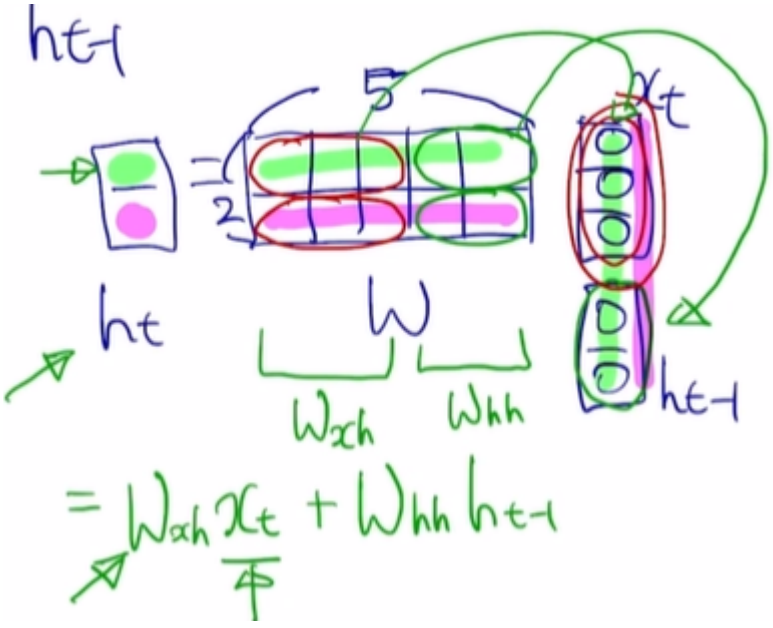


- t: 현재 타임스텝, w: weight

- h_{t-1}: old hidden-state vector
- x_t: input vector at some time step
- h_t: new hidden-state vector
- f_w: RNN function with parameters W
- y_t: output vector at time step t

- How to calculate the hidden state of RNNs
  - The state consists of a single "hidden" vector **h**

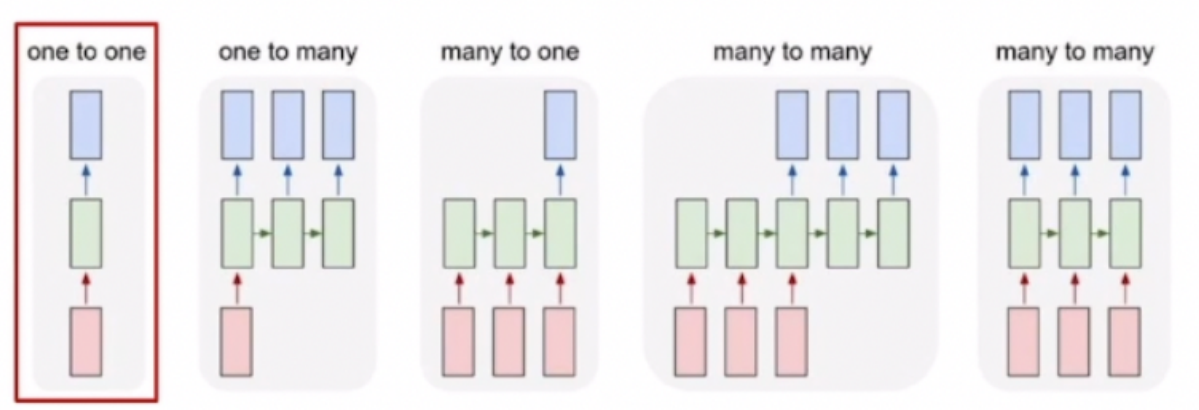




1. 위의 변수들에 대하여 $h_t = f_w(h_{t-1}, x_t)$의 함수를 통해 매 타임스텝나다 hidden state를 다시 구해준다.
2. 이때, W와 입력값 (x_t, h_{t-1}) 으로 tan_h를 곱해서 h_t 를 구해준다

3. 구해진 h_t, x_t 를 입력으로 y_t 값을 산출한다.

Types of RNN



one-to-one

- Standard Neural Networks
- 입출력의 timestep이 한 개
- sequence data가 아닌 경우
  - ex. [키, 몸무게, 나이]를 입력값이고 저혈압/고혈압인지 분류

one-to-many

- 입력은 하나의 timestep, 출력은 여러 개의 timestep
- Image Captioning
- 입력: 하나의 이미지 / 출력: 이미지에 대한 설명글

many-to-one

- 입력은 여러 개의 timestep, 출력은 하나의 timestep
- Sentiment Classification
- 입력: sequence of text / output: 긍부정 분류

many-to-many (1)

- Machine Translation
- 입력을 다 읽고 출력
- 입력: sequence of text / 출력: sequence of text
  - ex. 입력은 영어 문장, 출력은 번역된 한글 문장

many-to-many (2)

- 입력이 들어올 때마다 output 출력
- POS tagging
- Video classification on frame level
  - 영상을 프레임별로 분석/예측

Character-level Language Model

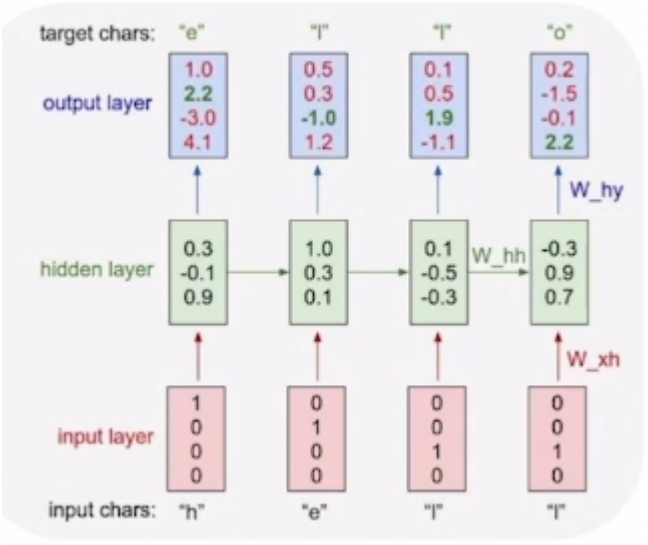
Character-level Language Model

Language model

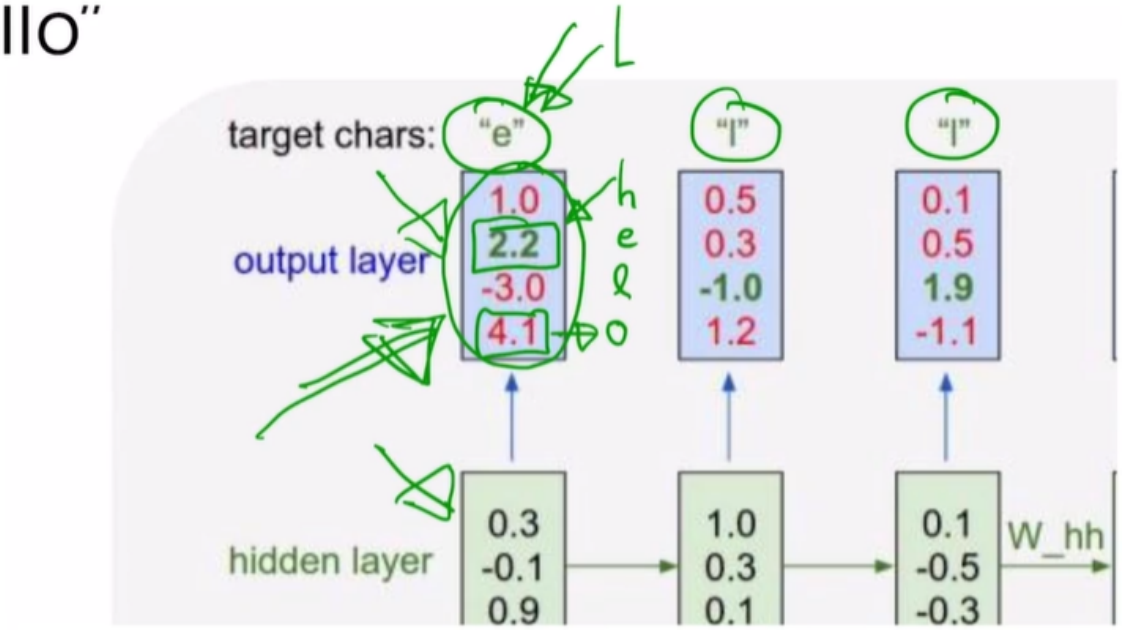
- 이전에 등장한 문자열을 기반으로 다음 단어를 예측하는 태스크

Character-Level Language Model

- 문자 단위로 다음에 올 문자를 예측하는 언어 모델
- example of ‘hello’



llo"



- 이때 각 타임스텝별로 output layer를 통해 차원이 **유니크한 문자의 개수**인 벡터를 출력하는데 이를 **logit**이라고 부르며, softmax layer를 통과시키면 one-hot vector 형태의 출력값이 나옴

언어 모델 예시

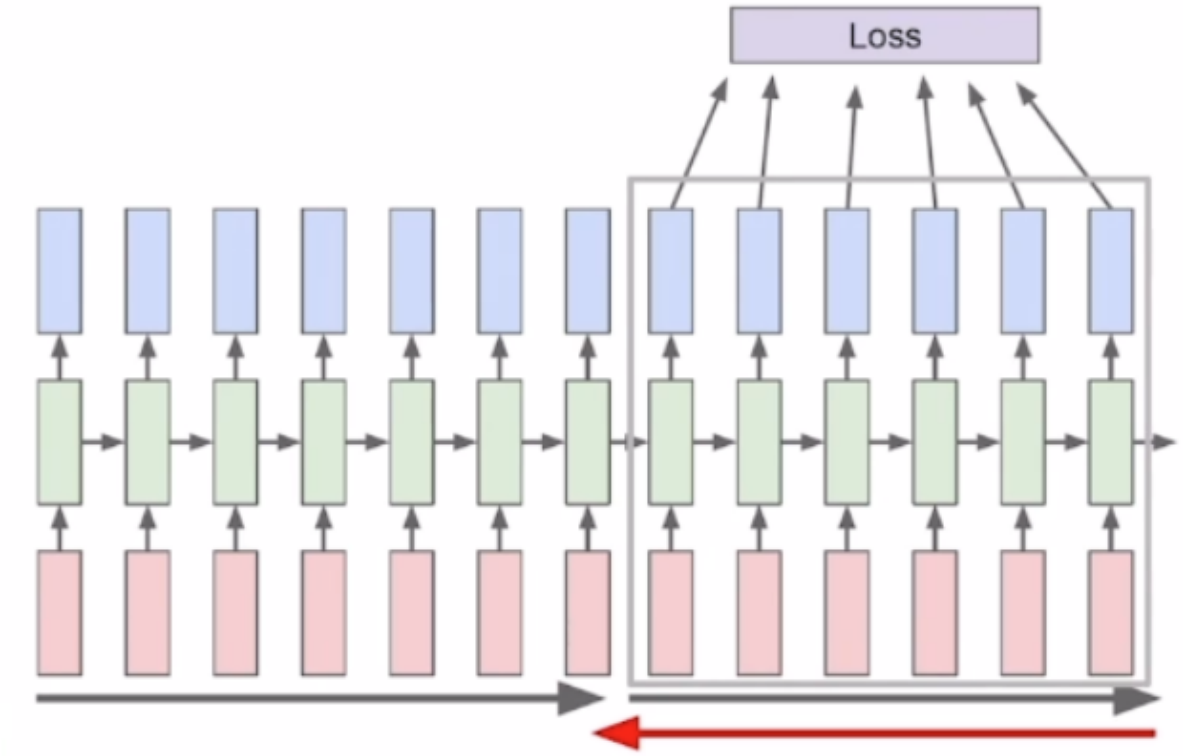
- 셰익스피어의 글을 활용해 더 많은 학습이 진행될수록 완전한 형태의 문장을 출력
- 전 타임스텝까지의 주식값을 활용해 다음날 주식값 예측

- 인물별 대사, Latex로 쓰여진 논문, C프로그래밍 언어 같은 경우도 학습을 통해 텍스트 생성할 수 있음

Backpropagation through time apendency

RNN모델 학습 방법: Truncation, BPTT

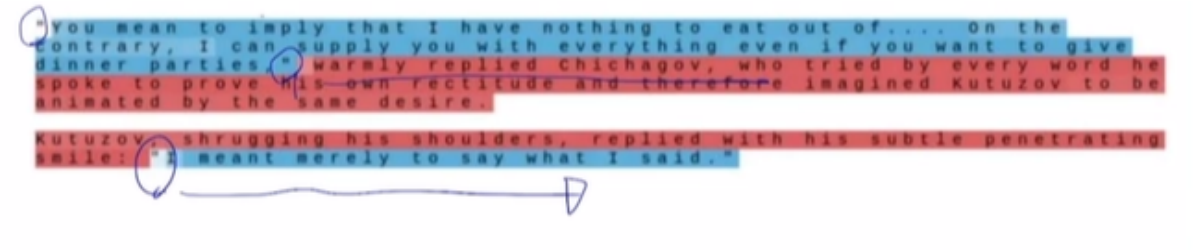
Truncation



- 제한된 메모리 자원 내에서 모든 시퀀스를 학습할 수 없기 때문에 일정 부분 잘라서 학습에 사용하는 것

BPTT

- Backpropagatoin through time
- RNN에서 타임스텝마다 계산된weight를 backward propagation을 통해 학습하는 방식



- 특정 cell의 학습 결과를 시각화한 모습

Vanishing/Exploding Gradient Problem in RNN

Toy Example

- $h_t = \tanh(w_{xh}x_t + w_{hh}h_{t-1} + b), t = 1,2,3$
- For $w_{hh} = 3, w_{xh} = 2, b = 1$

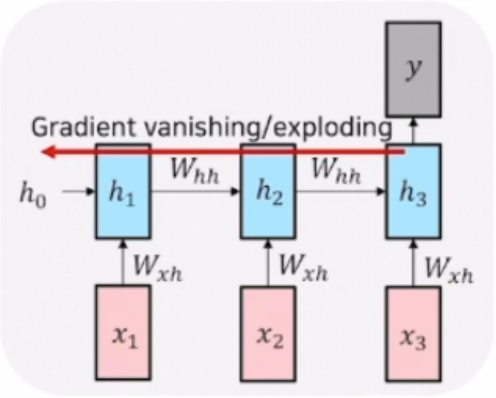
$h_3 = \tanh(2x_3 + 3h_2 + 1)$

$h_2 = \tanh(2x_2 + 3h_1 + 1))$

$h_1 = \tanh(2x_1 + 3h_0 + 1))$

...

$h_3 = \tanh(2x_3 + 3\tanh(2x_2 + 3h_1 + 1) + 1)$



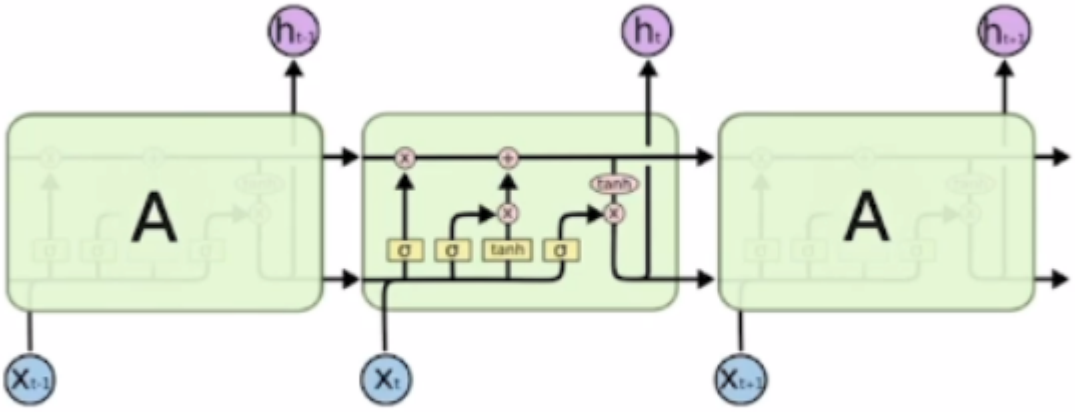
- 세 번째 time step의 hidden state인 h_3를 h_1으로 표현하면 $h_3 = tanh(2x_3 + 3tanh(2x_2 + 3h_1 + 1) + 1)$ 로 표현됨
- BPTT를 통해 gradient를 계산해주면, tanh로 감싸진 괄호안의 값들 중에 3 값이 속미분되어 나오게 됨
- time step 길이가 더 길어진다면 미분값이 기하급수적으로 커질 것이고, 속미분되어 나오는 W의 값이 1보다 작다면 미분값은 기하급수적으로 작아짐
- 따라서 Gradient Vanishing/Exploding 문제가 발생하고 이 문제가 Long-Term-Dependency를 일으키게 됨

LSTM

Long Short-Term Memory (LSMT)

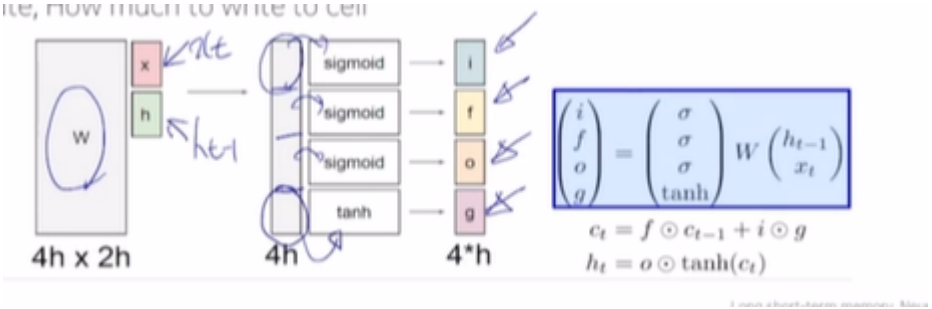
LSMT

- RNN의 long term dependency 문제 해결



The repeating module in an LSTM contains four interacting layers.

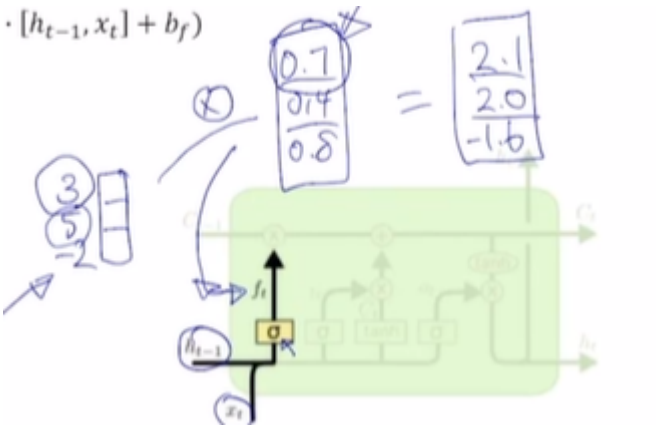
- 전 타임 스텝에서 두 가지의 정보가 들어옴
  - C_t, h_t
  - $C_t, h_t = LSTM(x_t, C_{t-1}, h_{t-1})$
- IFOG



  - I: input gate, Whether to write to cell / sigmoid
  - f: Forget gate, Whether to erase cell / sigmoid
  - o: Output gate, How much to reveal cell / sigmoid

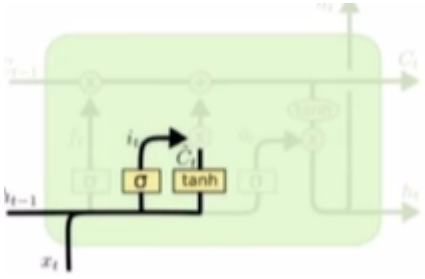
- g: Gate gate, How much to write to cell / tanh

Forget gate



- $f_t = \sigma(W_f \cdot [h_t, x_t] + b_f)$

Gate gate



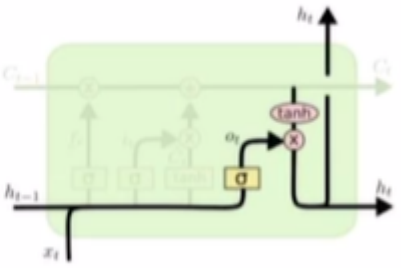
- Generate information to be added and cut it by input gate
  - $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$
  - $C_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c$
- Generate new cell stsate by adding current information to previous cell state

Hidden state vector

- Generate hidden state by passing cell state to tanh and output gate
- Pass this hidden state to next time step, and output or next layer if needed

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

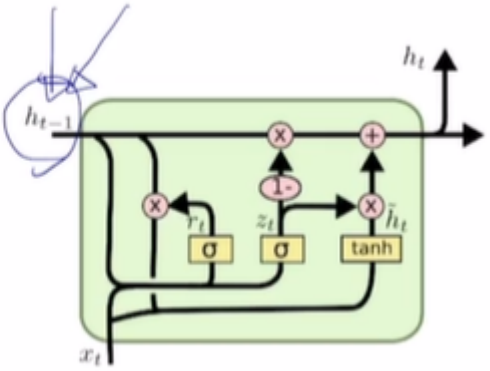
0~1 -1~1



Gated Recurrent Unit (GRU)

GRU

- What is GRU?
  - $z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$
  - $r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$
  - $\tilde{h}_t = \tanh(W \cdot [r_t \cdot h_{t-1}, x_t])$
  - $h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot \tilde{h}_t$
  - c.f) $C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$ in LSTM



- LSTM의 모델을 경량화해 적은 메모리를 사용하고 시간 단축
- cell state vector와 hidden state vector를 일원화해서 hidden state vector만 존재함