Lec 16. Gated Recurrent Units

Cho , Reset-gate , Update-gate

Will Cover

1. Introduction

- background - complex structure of LSTM
- introduction to 2gates - Reset , Update

2. GRU forward computation flow

- What is calculated at each gate (VS. LSTM ?)
- understanding flow as human language

3. GRU BPTT flow

- What to update ?
- how states are back propped (VS. LSTM ?)

4. Quick GRU tutorial

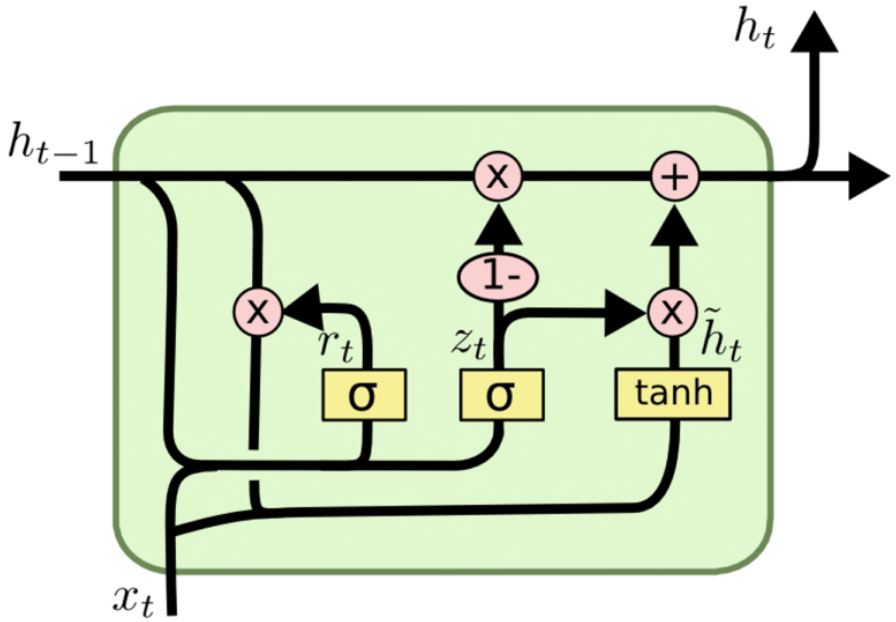
1. Introduction

background - Complex structure of LSTM

LSTM은 만능 킹왕짱일까? ⇒ 성능은 칭찬하나, 그에 비해 구조가 너무 복잡하다.

LSTM이 RNN의 고질적 기울기 문제 해결에 초점을 맞춰 개발되었다면, GRU는 그 목적은 계승하되 LSTM의 단순화를 목적으로 설계된 손한 신경 망이다.

introduction to 2gates - Reset, Update



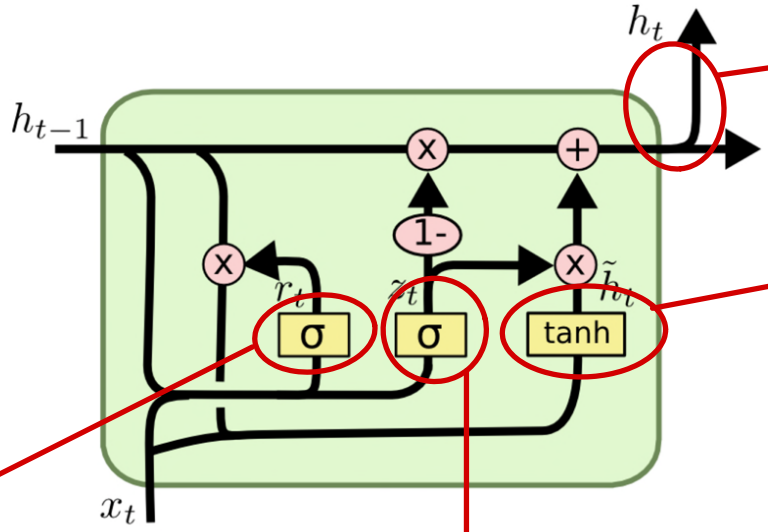
- 셀의 입/출력을 살펴보면 셀 사이 전달되는 정보는 h_t 하나!
- 2개의 Sigmoid activation ! = 2개의 gate !!
- Z_t 라고 되어있는 부분을 보면 2 군데에 적용되는 것을 확인 가능 !!

2. GRU forward computation flow

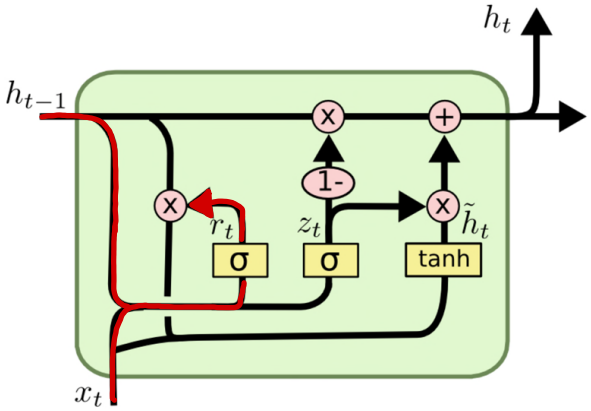
What is calculated at each gate (VS. LSTM ?)

understanding flow as human language

각 부분에서 어떤 계산이 이루어 지는지, 무엇을 의미하는지 알아보자!!



Reset Gate



$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$$

이전 hidden 정보와 현재의 입력을 살펴보고,

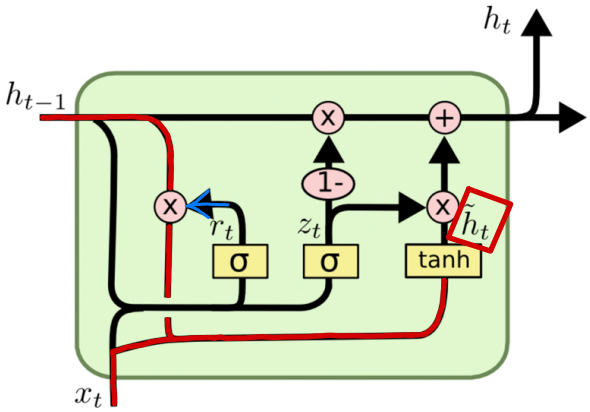
이번 셀의 분석결과를 계산하는데 h_{t-1}을 얼마나 사용할 것인지 양을 정한다!!

※ forget gate 와는 주는 영향이 다르다!!

σ 결과가 상태 정보에 곱해지고 끝나는 것이 아니라,

곱해진 상태 정보가 다음 연산에 활용됨!!

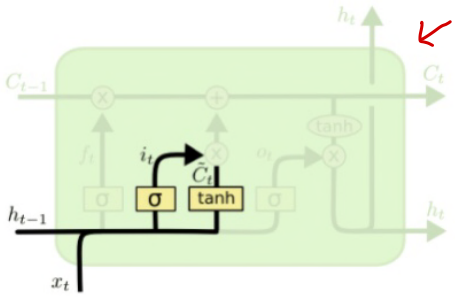
Candidate



$$\tilde{h}_t = \tanh\left(W_h \cdot [r_t * h_{t-1}, x_t]\right)$$

reset gate 의 결과를 이용하여

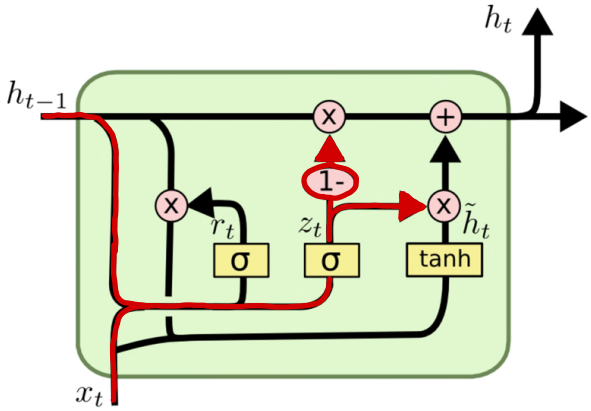
이번 셀의 출력이 될 1차 출력 정보를 생성하는 경로.



LSTM 구조에서는 1차 출력 데이터 연산에 바로 쓸 수 있는 h_t 가 존재 했지만 GRU 셀은 h_t 가 C_t, h_t 두 역할을 모두 수행하는 개념이기에 C_t 에서 현재 셀 연산에 쓰일 h_t 를 뽑는다는 느낌으로 $r_t * h_t$ 의 수행결과와 x_t 를 통해 $\tilde{h}_t$ 를 계산하는 단계 !

※ '*' 기호는 element-wise mul 를 의미하며 그 놈의 아다마르 곱임.

Update Gate



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

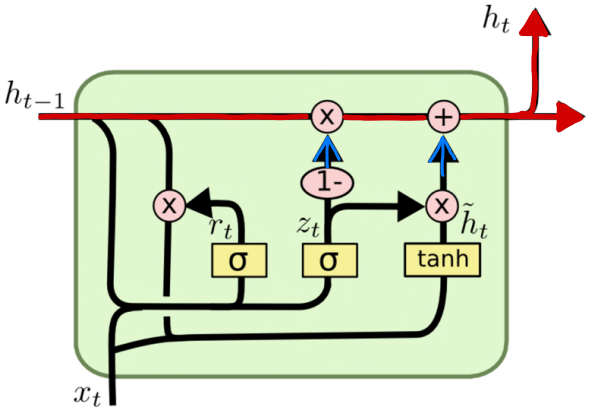
개인적인 생각으로는 GRU 의 핵심 단계 !!

지금까지 봤던 다른 게이트들과 마찬가지로 h_{t-1} 과 x_t 를 통해 Sigmoid 연산후 0 ~ 1 사이 게이트 값을 도출한다.

이 값이 어디에 적용되는지 살펴보면
오른쪽으로는 Candidate step 으로부터 계산된
1차 출력물에 곱해지고, 위쪽으로는
$(1-z)$ 값으로 적용되어 이전 상태 정보
h_{t-1} 에 적용되는 것을 알 수 있다.

※ 즉 다시 말하면 이전의 정보는 얼마나 유지할지,
현재 계산된 정보는 얼마나 취할지,
lstm 의 input & forget gate 의 역할을 Update gate 에서
모두 수행하는 것이다!!

New state



$$h_t = (1-z_t) * h_{t-1} + z_t * \tilde{h}_t$$

두 Gate 와 Candidate 과정의 결과가
최종적으로 new state (output) 으로
갱신되는 과정은 LSTM 과 흡사하다.
LSTM 에서의 C_t 계산 식과
비교해보면 더 그 유사함을 알 수 있다.

※ $C_t = f_t * C_{t-1} + i_t * g_t$

$h_t = (1-z_t) * h_{t-1} + z_t * \tilde{h}_t$