RNN 알고리즘은 반복적이고 순차적인 데이터(Sequential data)학습에 특화된 인공신경망의 한 종류로써, 내부의 순환구조가 들어있는 특징이 있다. 순환구조를 이용하여, 과거의 학습을 Weight를 통해 현재 학습에 반영한다. 기존의 지속적이고 반복적이며, 순차적인 데이터학습의 한계를 해결하였다. 현재의 학습과 과거의 학습의 연결이 가능해 졌으며, 시간에 종속된다. 음성 Waveform을 파악하거나(Machine Translation) , 텍스트의 문장 앞 뒤성분을 파악할 때(Named Entity Recognition), 주로 사용된다. 이 RNN 단점은 처음시작한 Weight의 값이 점차 학습이 될 수록 상쇄 된다는 것이였는데, 이를 보완한 알고리즘이 LSIM(Long Short Term Memory Network) 알고리즘이다.

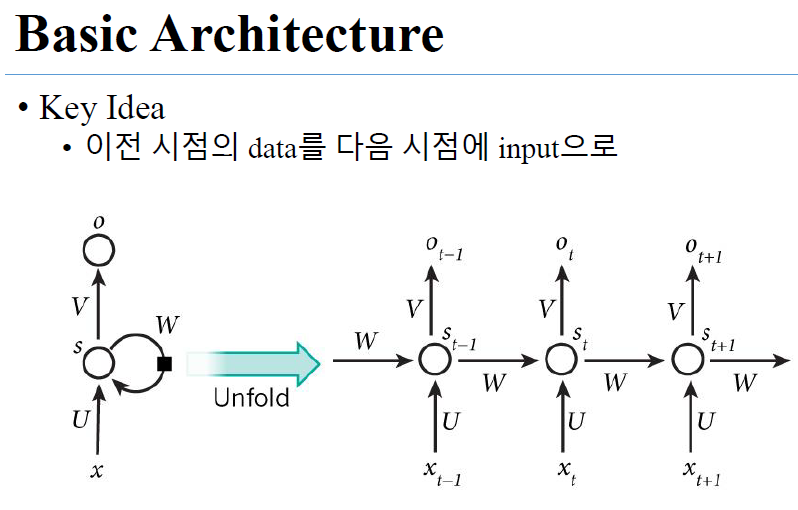
**Sequential data**

Text, Audio, Video 등 연속된 데이터

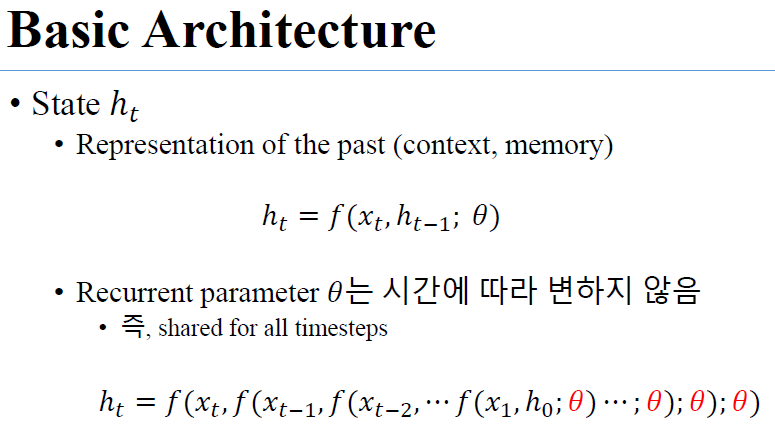
시간에 따라 바뀜

이전 데이터가 다음 데이터에 영향을 끼침

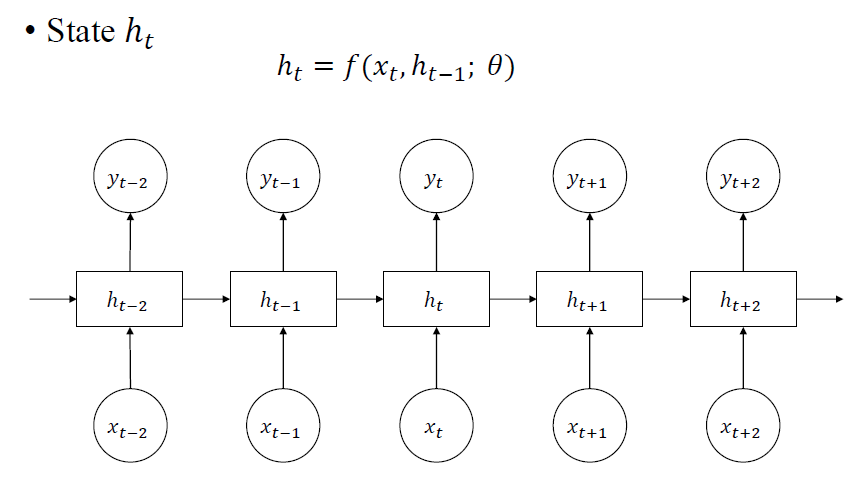
문맥, 메모리를 모델링하는 게 필요함



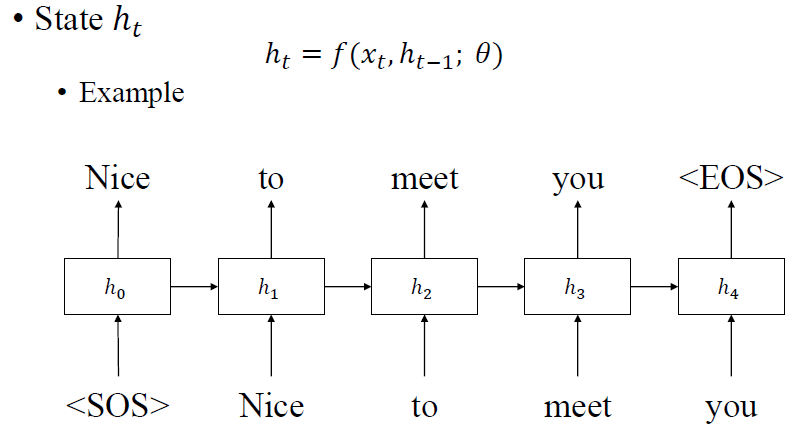
O: output x: input s: hidden state(layer) w: 이전 데이터를 입력



세타 θ는 시간에 따라 변하는 게 아님(모든 파라미터를 모은 것이 세타, b나 W, U가 세타)

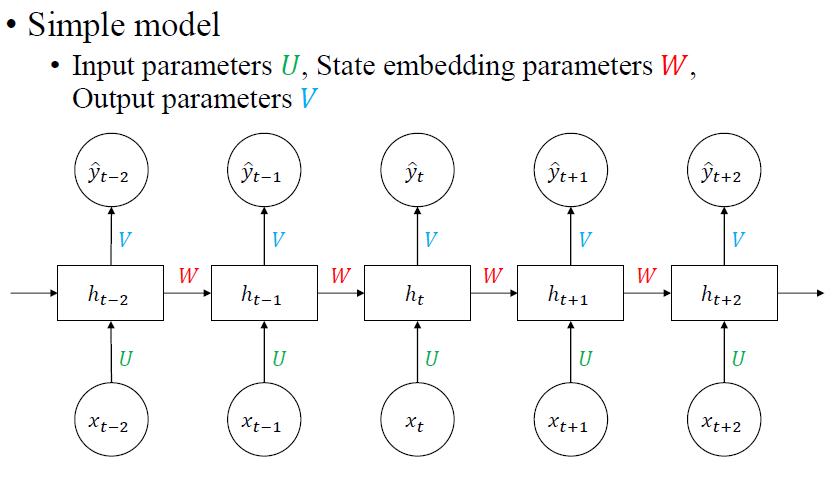


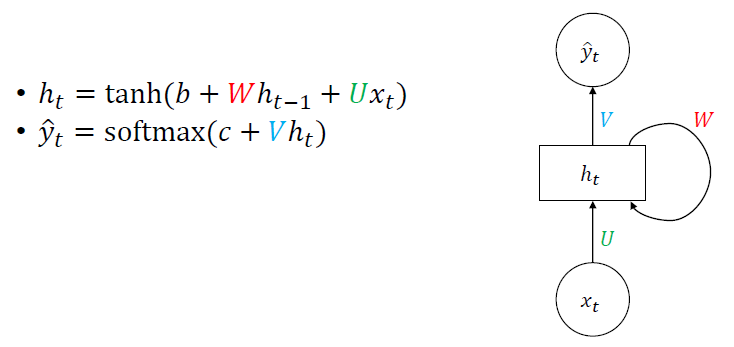
계산되는 방식이 state가 진행할 때에도 같다.



Start of Sequence(SOS)

End of Sequence(EOS)





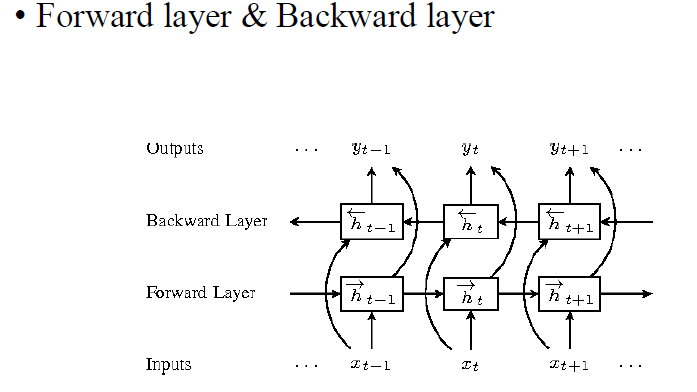
b나 W, U가 세타

Input x에 곱해줄 u

히든 스테이트 h에 곱해줄 w

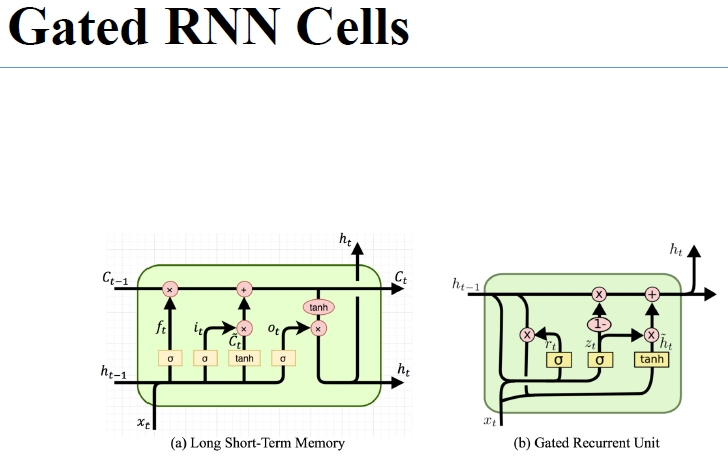
b는 bias

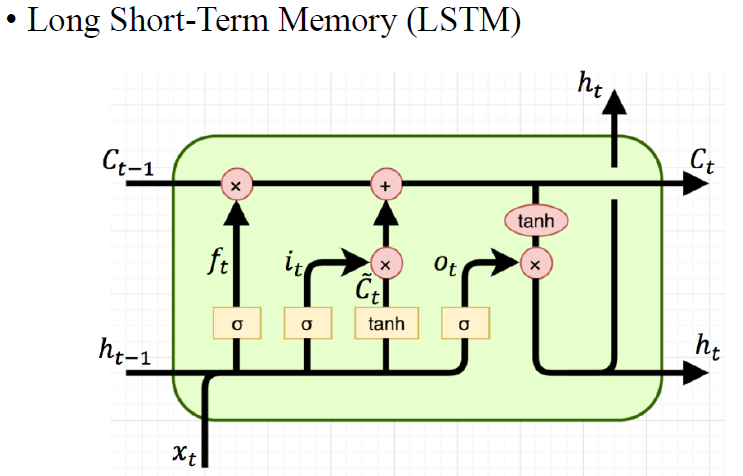
파라미터가 초기화 되면 변하지 않은 것이 아니라 한 문장에 대해 fix가 되고 그 문장을 학습시켜 나중에 업데이트가 된다. 한 문장안에서 시간이 흐르는 동안은 고정된다.



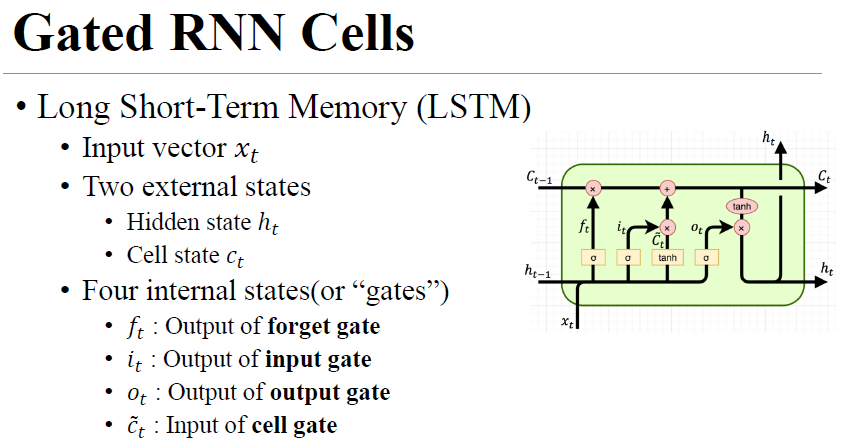
Forward layer 시간 흘러가는 방향

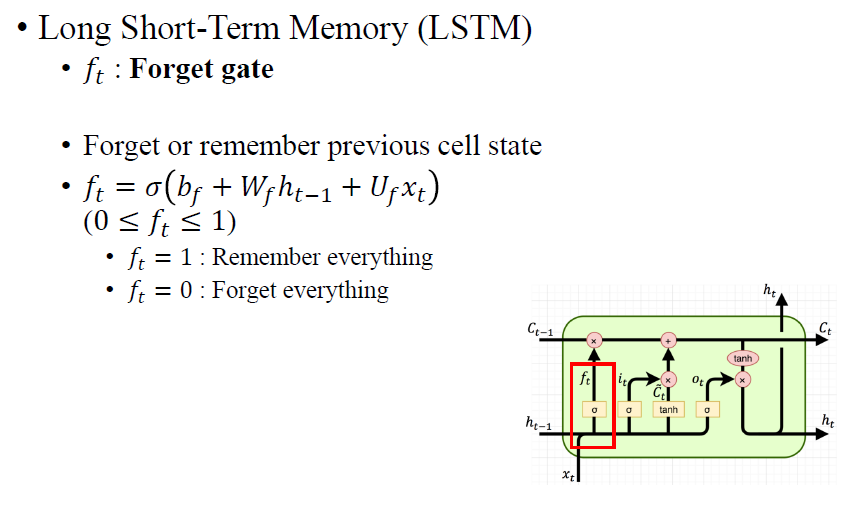
Backward layer 시간 반대로 흘러가는 방향





σ는 sigmoid



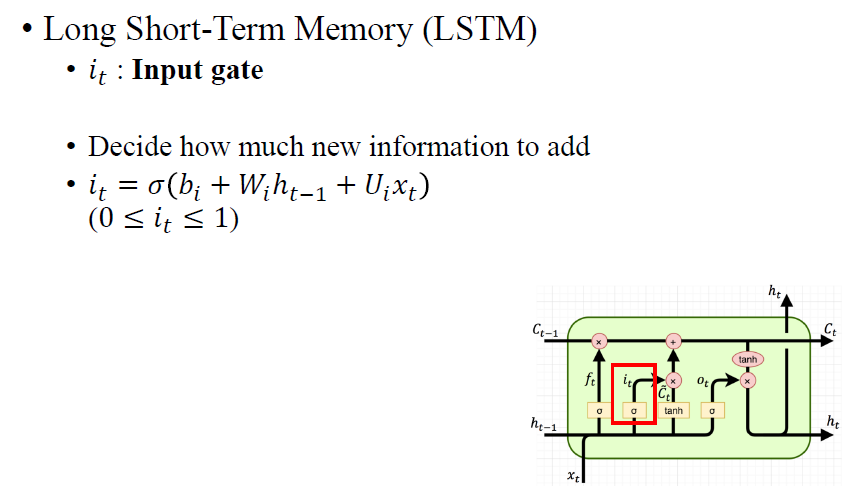


빨간색 박스 부분이 forget gate, ft=각각의 weight를 곱해주고 bias를 더해 줌

Ft=1이면 이전 state는 기억하겠다

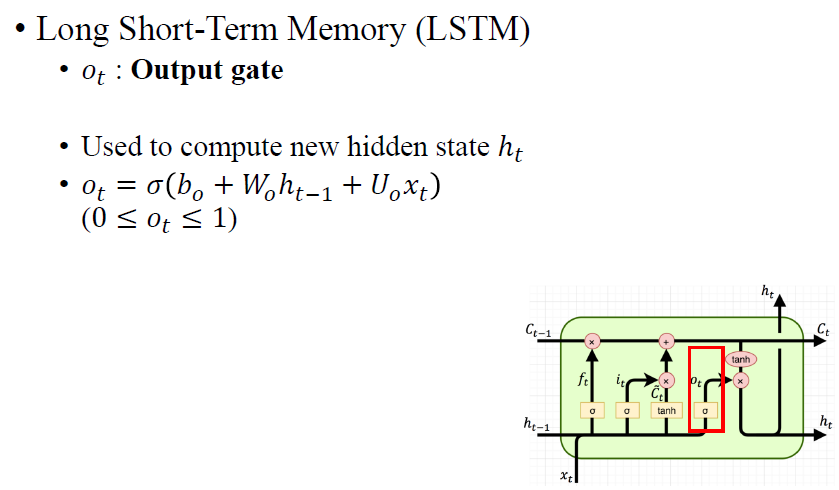
Ft=0이면 이전 state는 기억하지 않겠다 지금 state로 의존하겠다.

1에 가까울수록 이전state를 많이 사용하겠다, 0에 가까울수록 이전 state를 사용하지 않겠다.

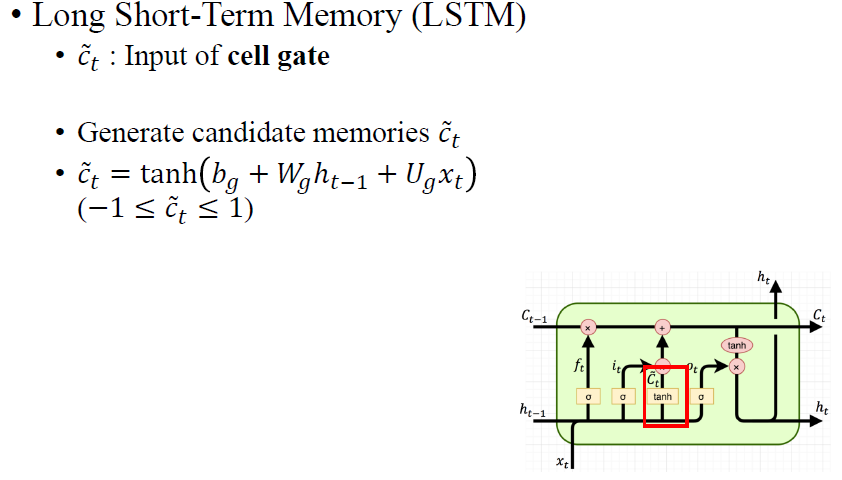


it는 이번에 들어온 새로운 정보를 얼마나 살릴지 무시할 지 정함

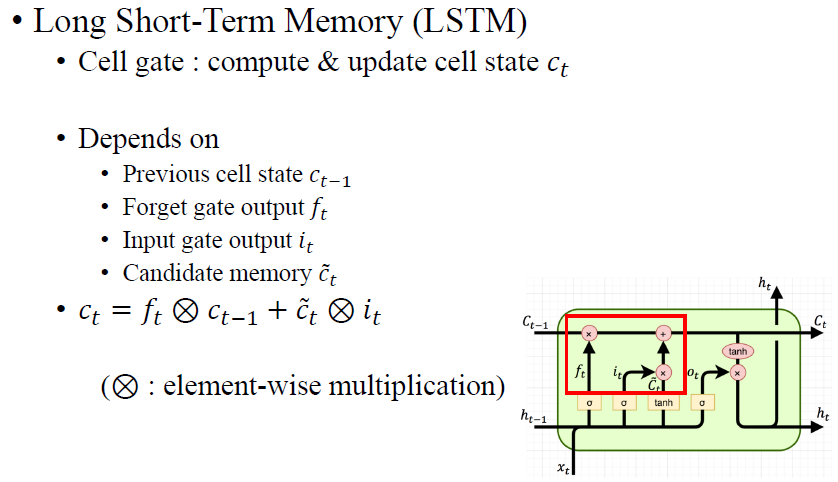
forget gate와 input gate의 weight는 다른 것.



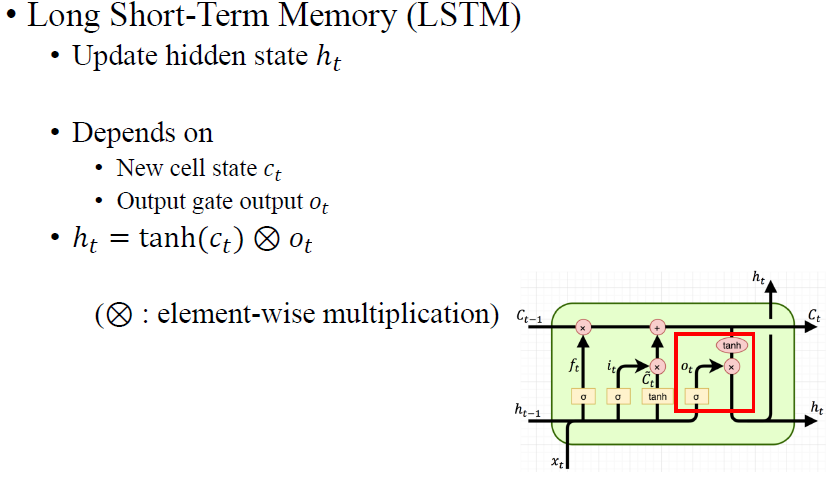
생긴 것은 똑같지만 하는 역할이 다르기 때문에 파라미터의 값도 bo, bi, f가 다르고 각각 다른 방향으로 학습이 된다.



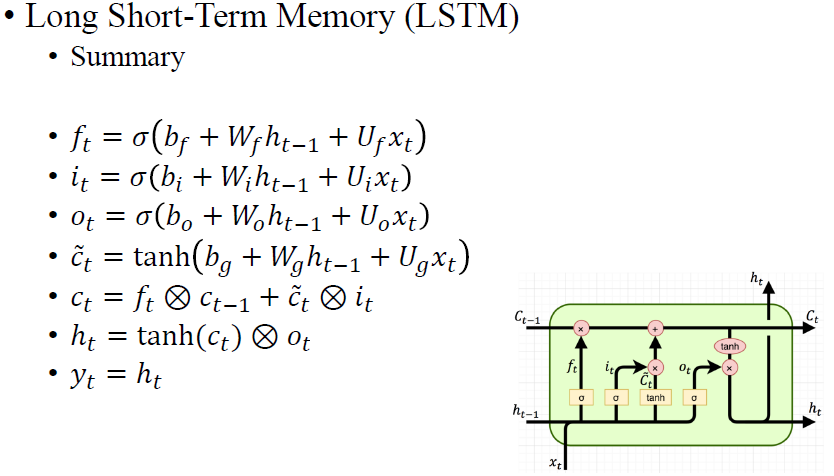
유일하게 sigmoid가 아님 tanh이다. 새로운 cell state를 계산하기 위한 현재 들어온 정보(xt)를 담고 있는 하나의 후보



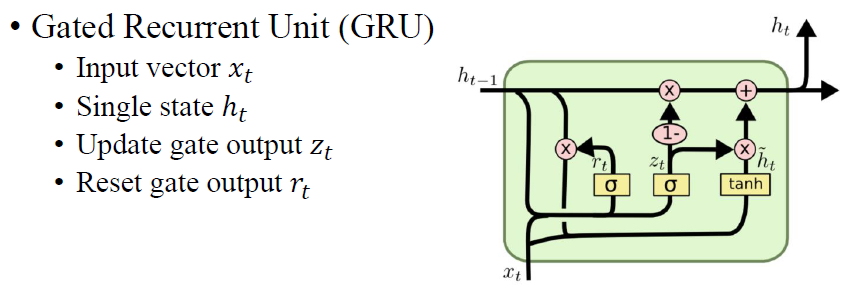
빨간색 네모 박스에서 ft와 ct-1을 곱하고 ct와 it를 곱해서 더해준다.



교재에 ot-1은 오타 🡪 ot



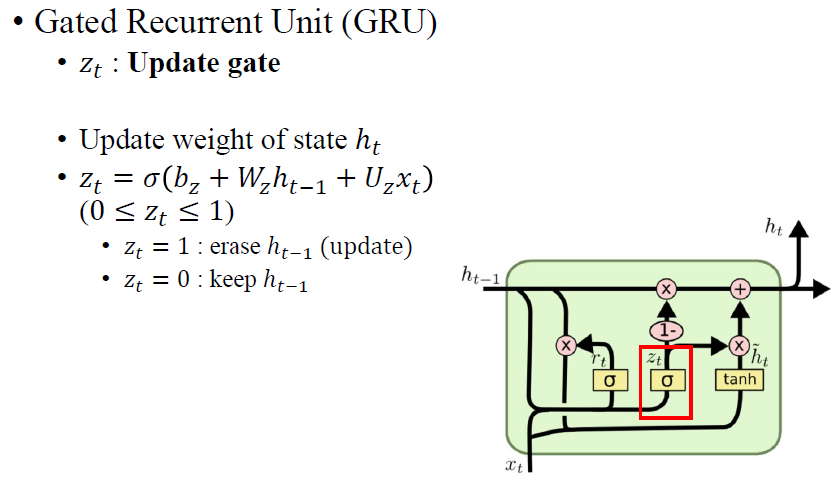
Bias가 각각 따로 있는 것



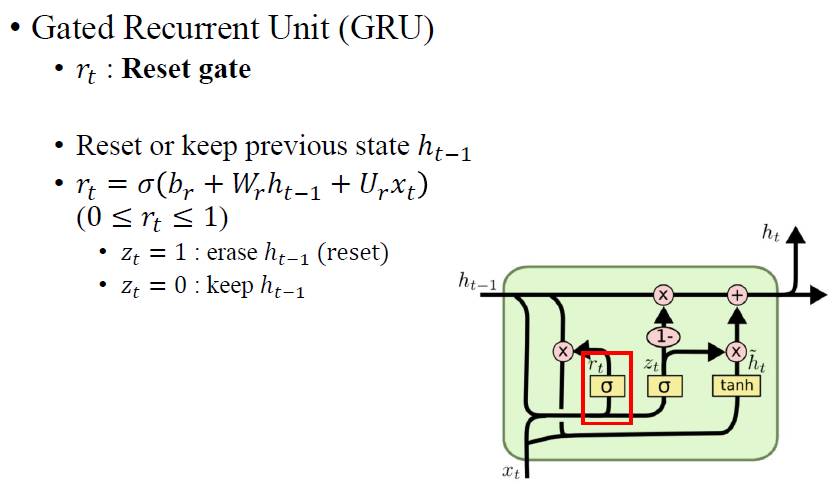
2014년 한국인 교수님이 고안

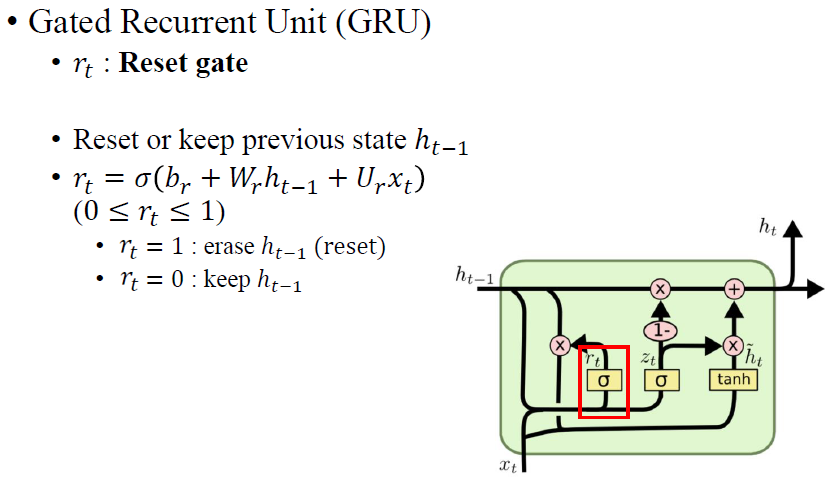
LSTM과 비교하면 히든 스테이트가 1개

계산되는 값을 LSTM보다 적지만 계산이 더 복잡

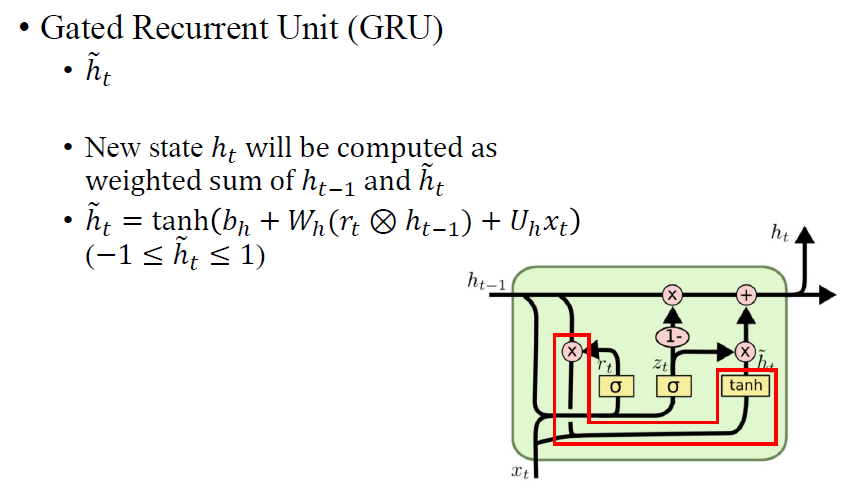


Zt가 1에 가까울수록 ht-1을 지우고 새로 업데이트 하고 0에 가까울수록 유지한다.



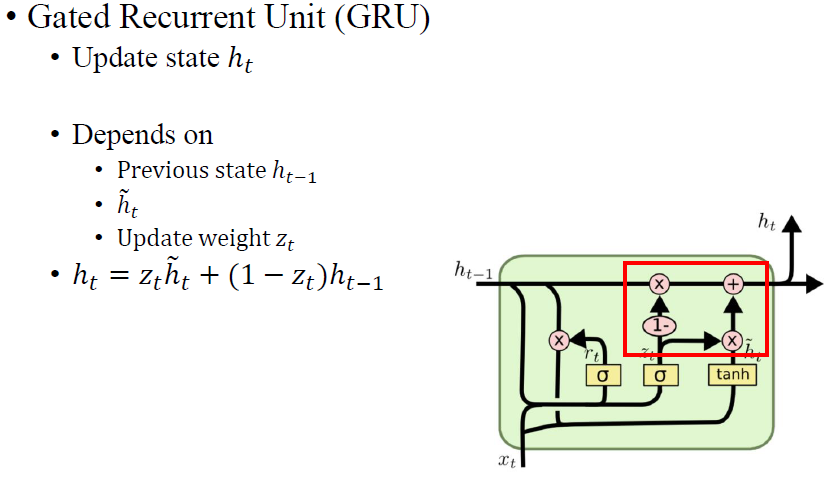
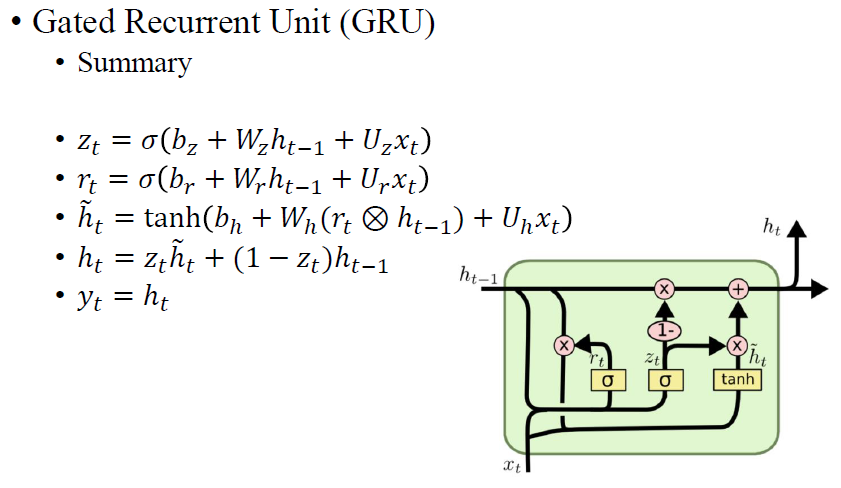


교재는 zt라고 되어있는데 오타

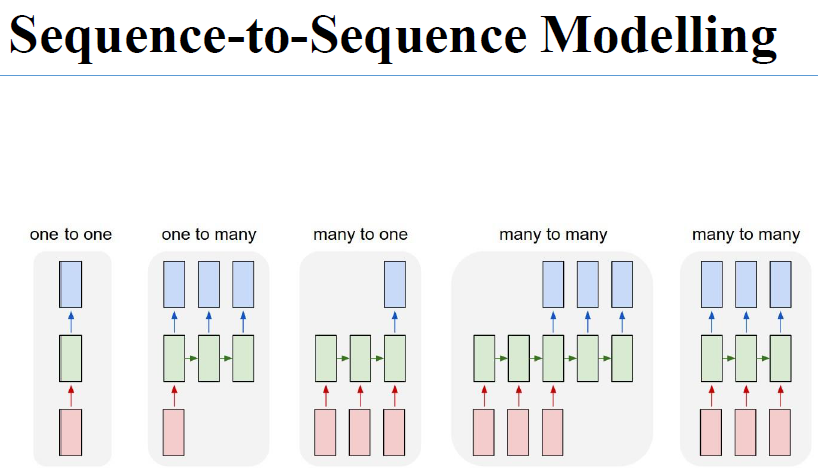


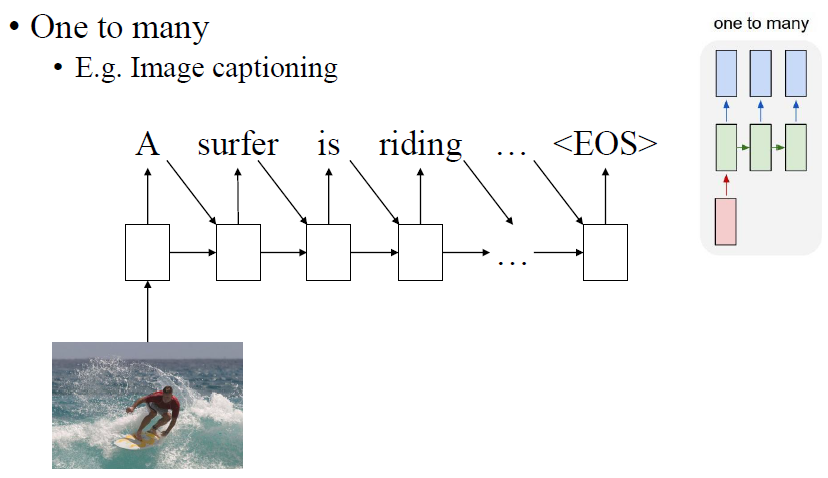
식을 보고 그림을 이해할 것

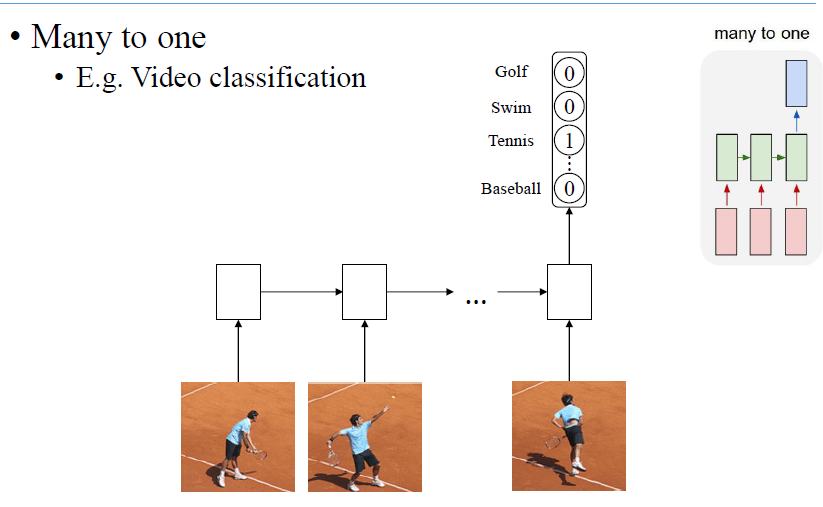
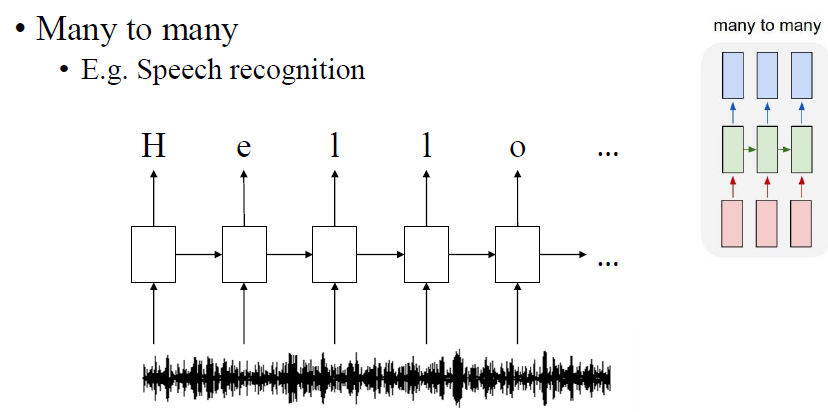
Rt와 ht-1을 곱해준 것과 bias와 xt에 tanh한 것

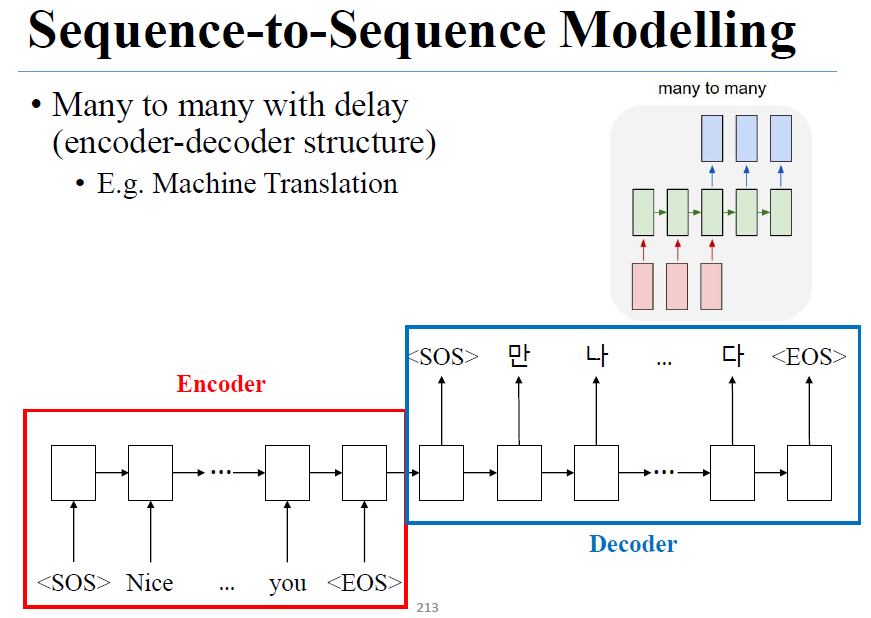
 

이러한 Cell을 통해 어떠한 것을 할 수 있는지 🡪 sequence-to-sequence modelling







Many to many with delay는 번역시스템에 주로 사용됨

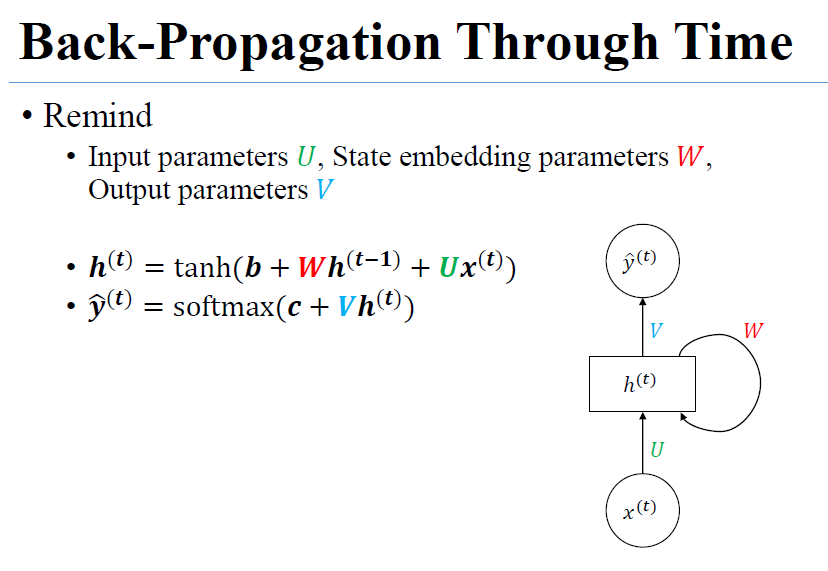
Training : Teacher-forcing

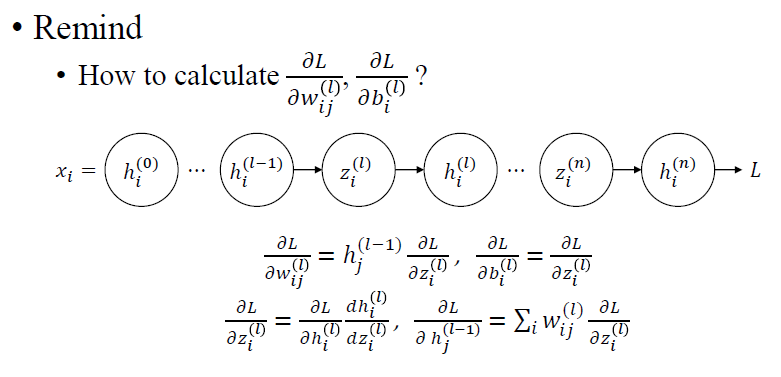
트레이닝 단계에서 정답을 넣어 줌

SOS에서 hidden state를 거쳐서 예를 들어 Nice to meet you라는 문장을 만들어야 하는데 dog라는 결과 값이 나왔다고 했을 때 선생님처럼 dog가 아니라 nice가 들어가야한다고 알려줌

Generation

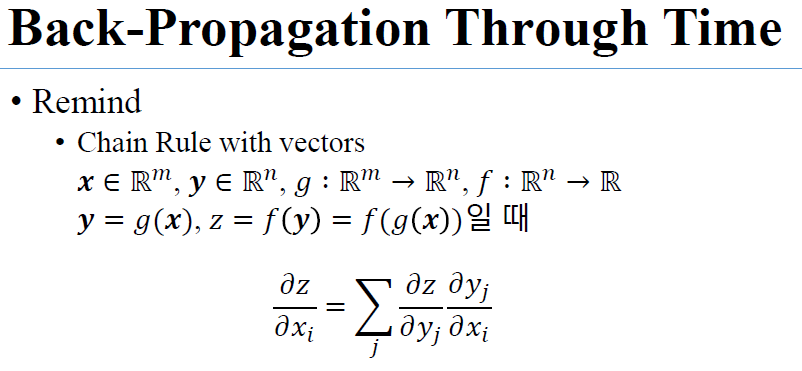
정답이 없기 때문에 SOS에서 hidden state를 거쳐 결과 값을 다음 레이어에 넣어 줌

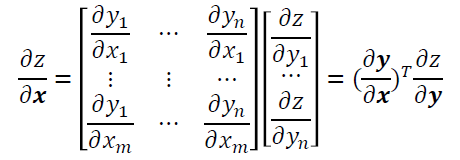
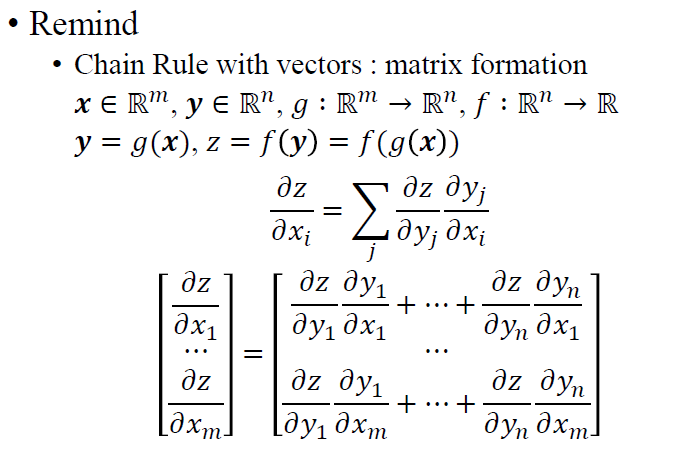


Loss를 최소화하는 것이 목적 🡪 함수가 있을 때, 미분을 하여 기울기를 통해 설정

미분은 activation function이 어떤 것인지에 따라 미분 🡪 뒤에서부터 앞단까지 쭉 미분

TensorFlow에서 이러한 미분연산을 다 해 줌 🡪 이러한 과정은 증명일 뿐





행렬 벡터 곱의 형태로 사용하겠다

