

x

RNN

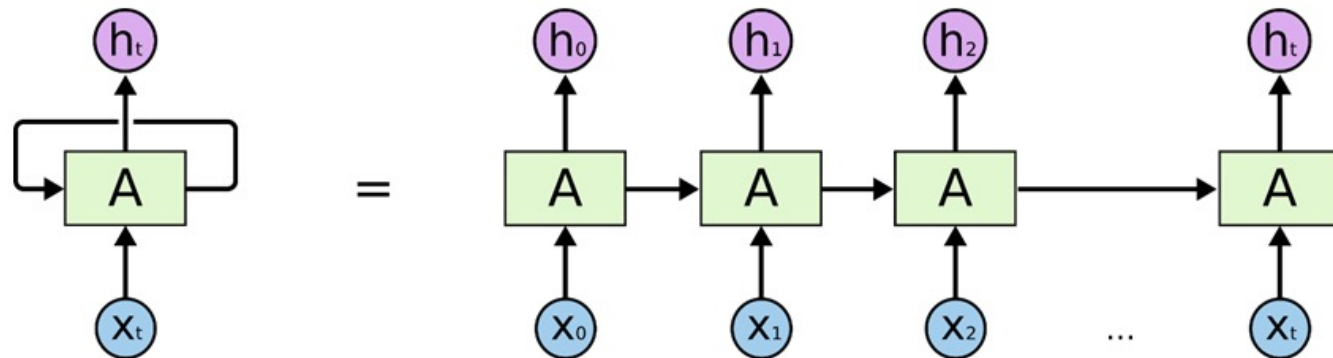
16기 문예진

RNN이란?



Recurrent Neural Network 순환 신경망

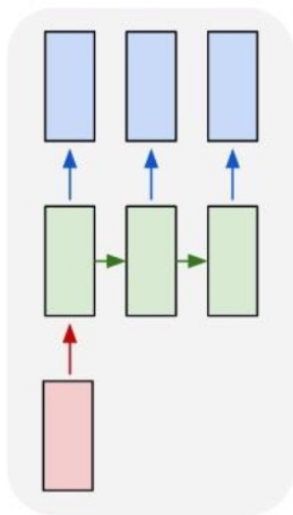
- 은닉 계층 안에 하나 이상의 순환 계층을 갖는 신경망
- Sequential 데이터의 “순서”가 고려되어야 할 경우, 순서를 유지하며 학습



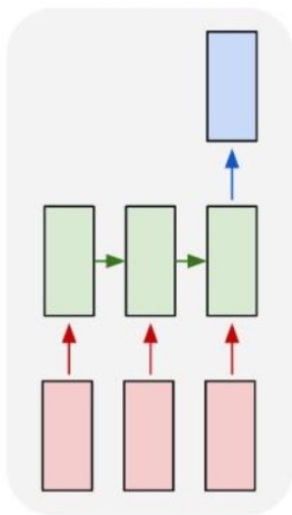
RNN task 종류



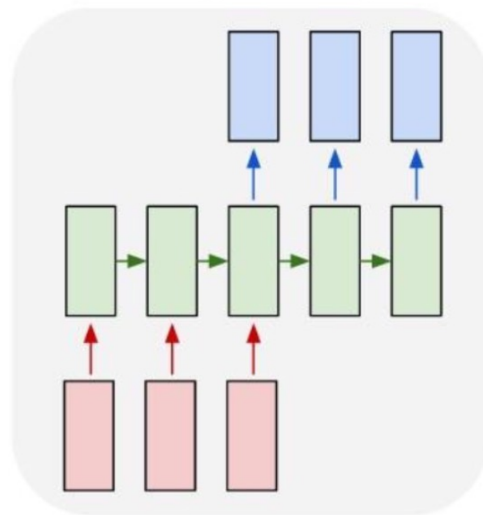
one to many



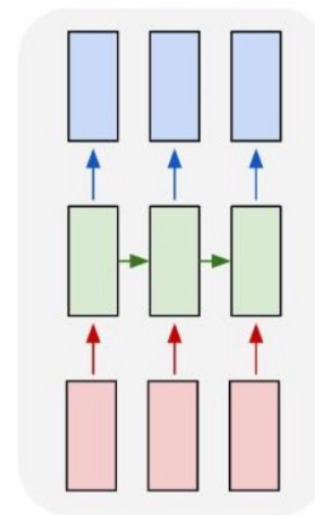
many to one



many to many



many to many



RNN task 종류

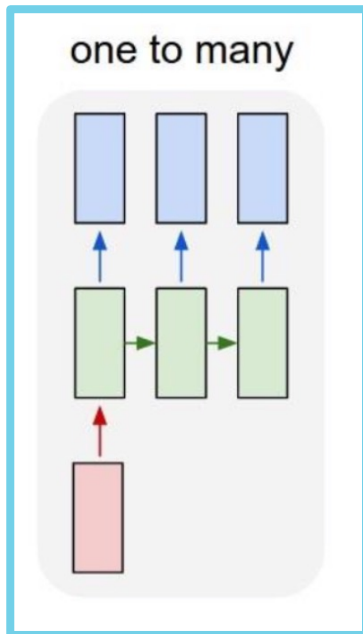
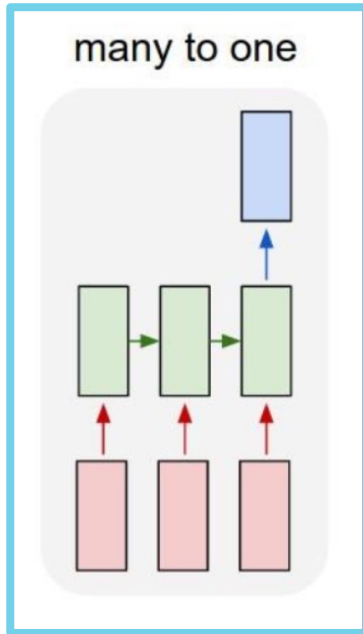


Image Captioning

- one : image vector
- many : sequence of words

<p>A young boy is playing basketball.</p> 	<p>Two dogs play in the grass.</p> 	<p>A dog swims in the water.</p> 
<p>A group of people walking down a street.</p> 	<p>A group of women dressed in formal attire.</p> 	<p>Two children play in the water.</p> 
<p>A skier is skiing down a snowy hill.</p> 	<p>A little girl in a pink shirt is swinging.</p> 	<p>A dog jumps over a hurdle.</p> 

RNN task 종류



Sentiment Classification

- many : sequence of words
- one : sentiment(좋/싫)

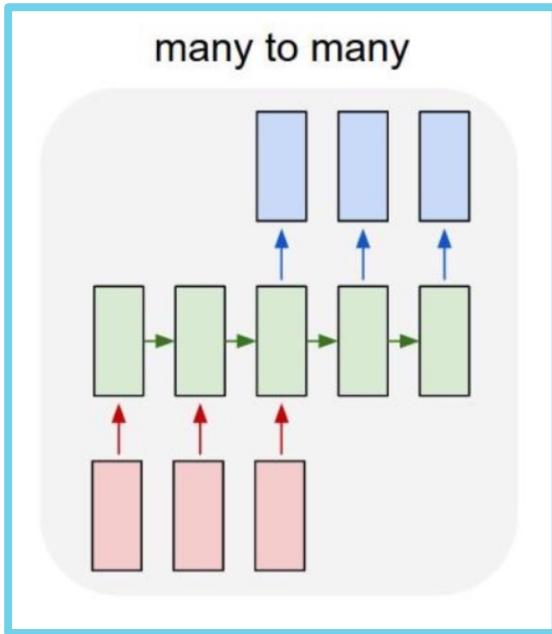
"I love this movie.
I've seen it many times
and it's still awesome."



"This movie is bad.
I don't like it it all.
It's terrible."

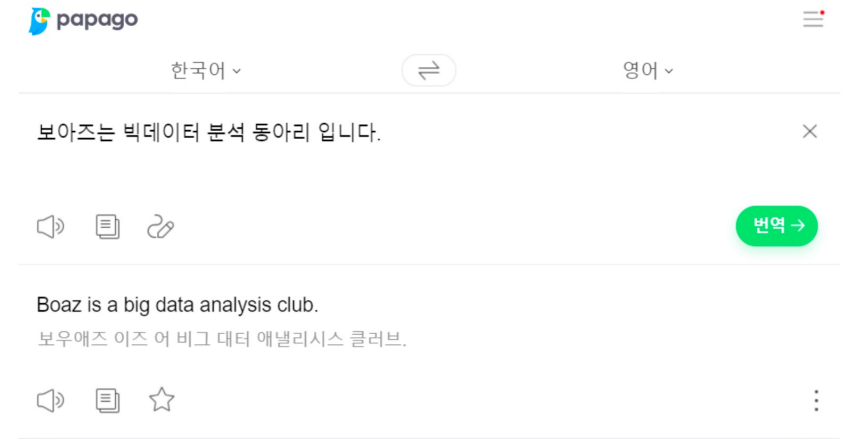


RNN task 종류

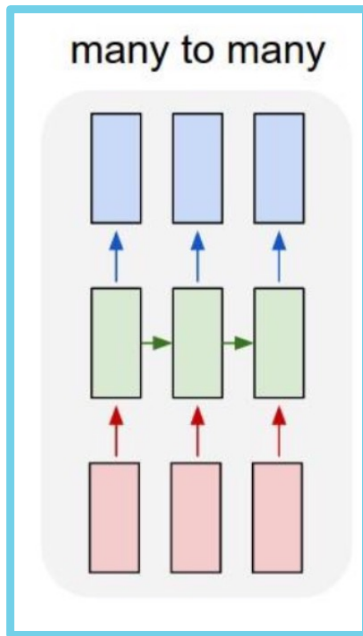


Machine Translation

- many : sequence of words
- many : 단어 순차 데이터(문장)
- “Sequence to Sequence”



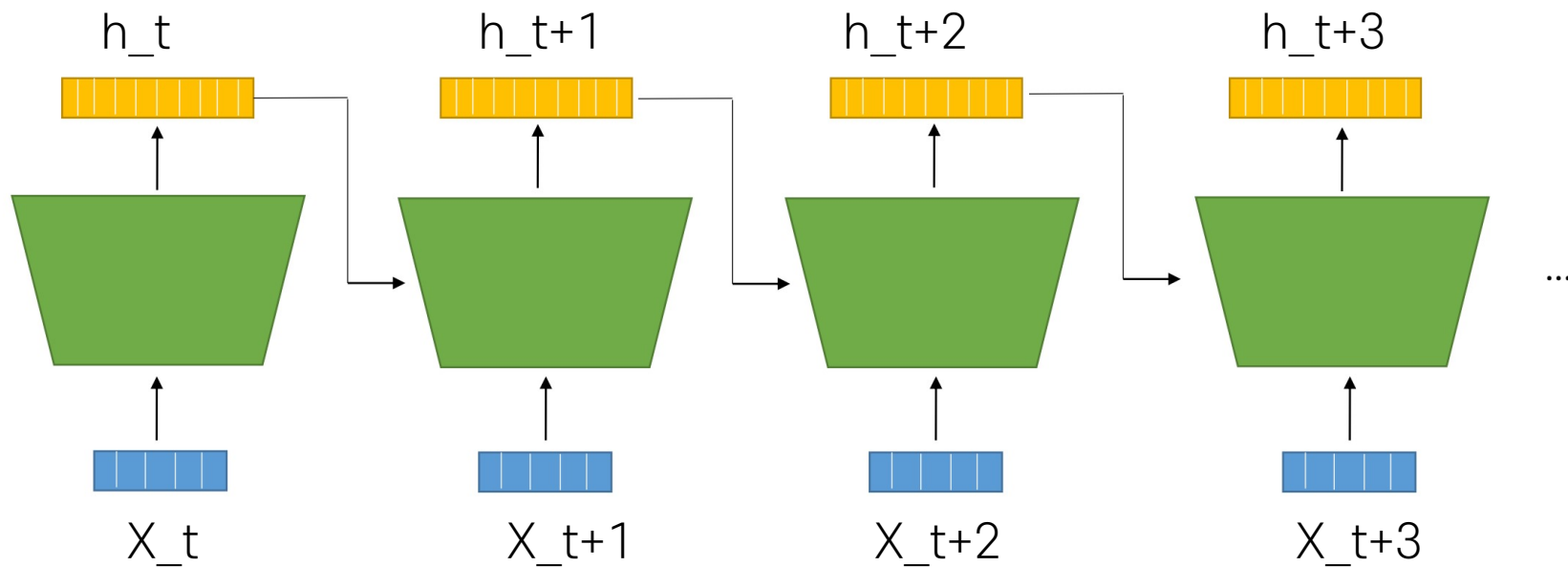
RNN task 종류



Noun Classification

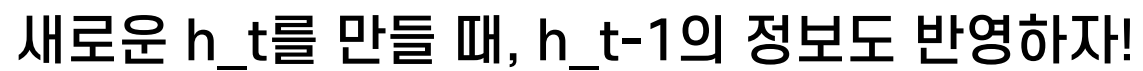
- many : sequence of words
- many : classifier of each word

RNN 작동원리



새로운 h_t 를 만들 때, h_{t-1} 의 정보도 반영하자!

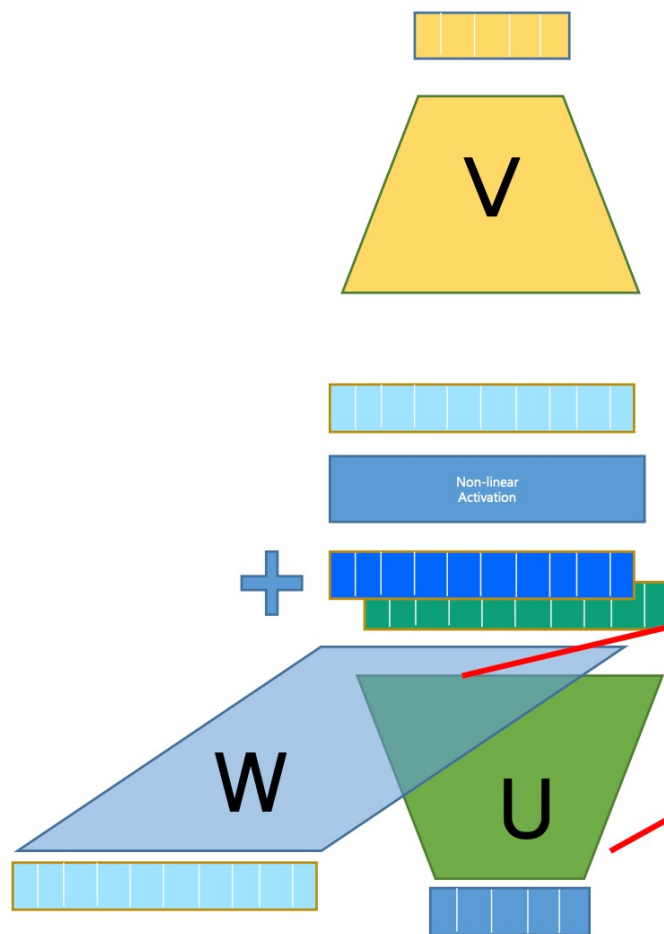
A stylized illustration of a city skyline. It features several skyscrapers of varying heights and widths, some with horizontal lines representing windows. To the left, there is a Ferris wheel. The background is a solid blue sky, and the foreground is a solid orange ground. The style is simple and graphic.



RNN 작동원리



수식표현

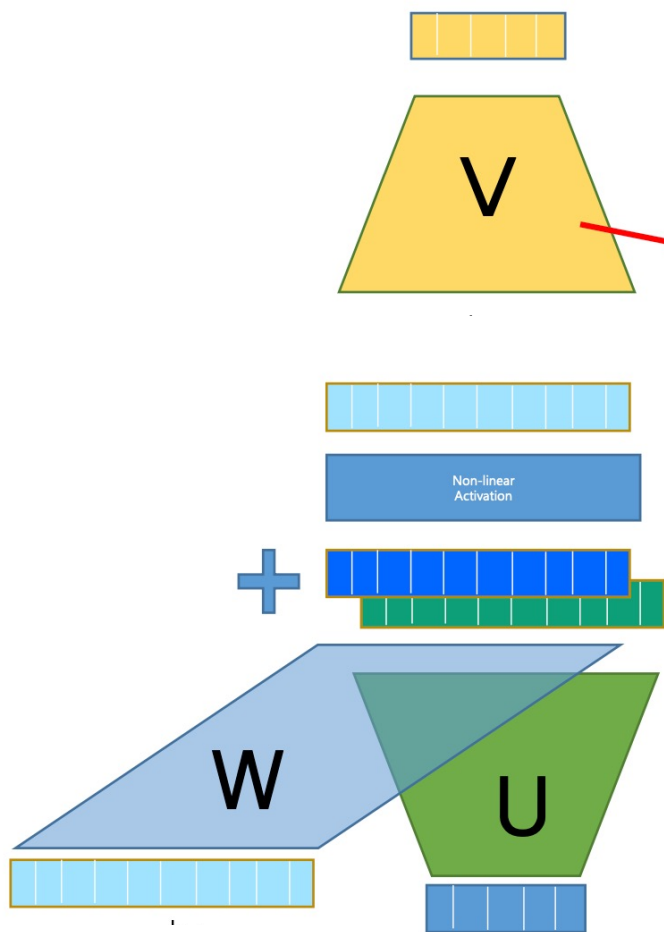


$$h_t = f(Ux_t + Wh_{t-1})$$

RNN 작동원리



수식표현



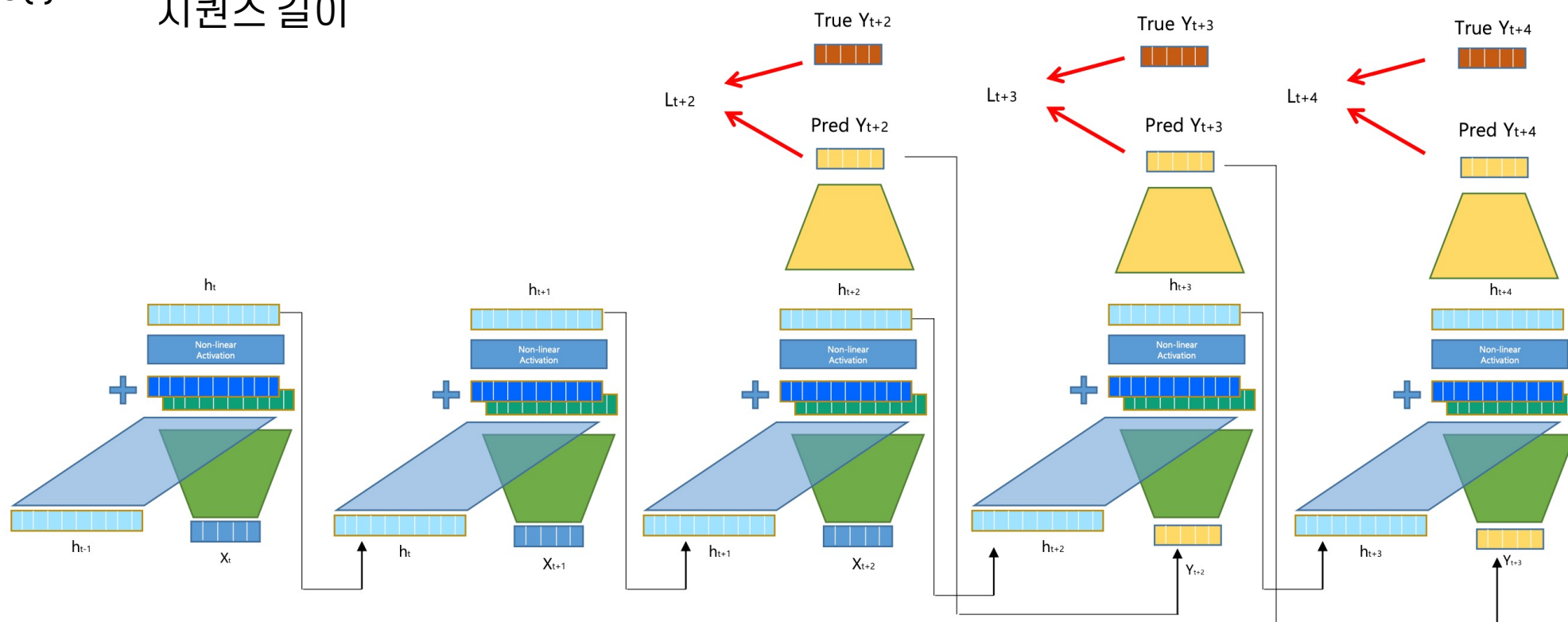
$$h_t = f(Ux_t + Wh_{t-1})$$

$$y_t = f(Vh_t)$$

RNN 작동원리



$$\text{Loss}(i) = \frac{\sum \text{loss}(\text{true } Y_t, \text{pred } Y_t)}{\text{시퀀스 길이}}$$

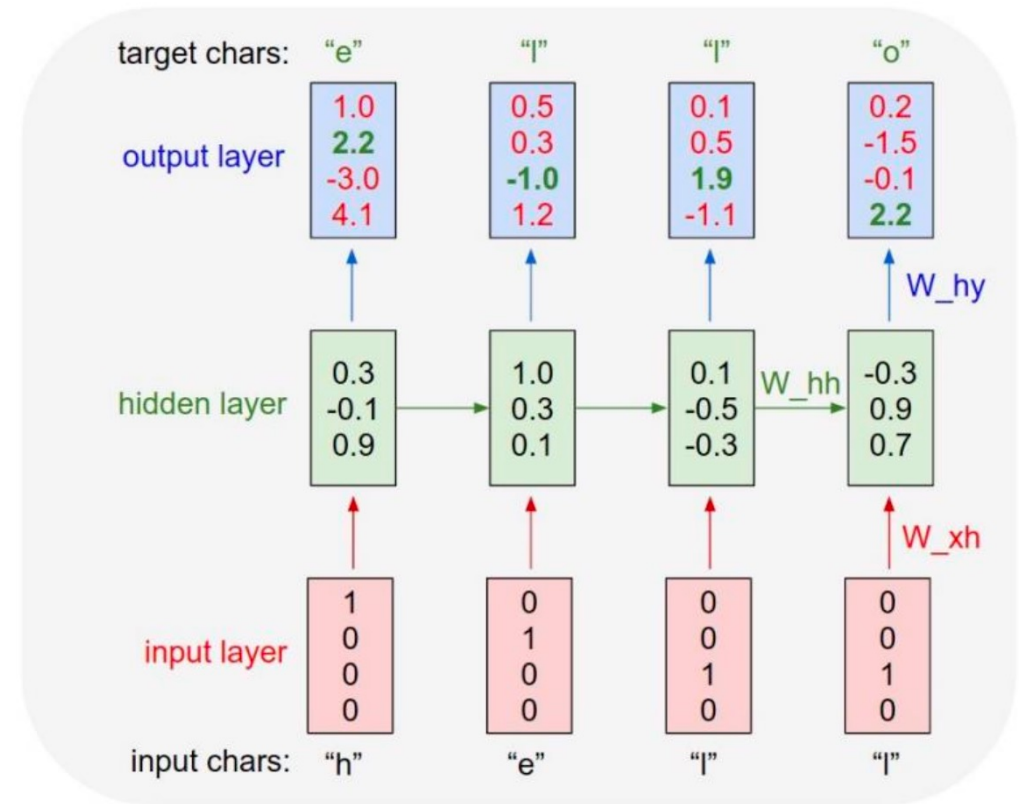


RNN 사용 예시



Language Model

- $X_t = \text{hello} \rightarrow [h, e, l, o]$
- 글자별로 one-hot encoding
 - $h : [1, 0, 0, 0]$
 - $e : [0, 1, 0, 0]$
 - $l : [0, 0, 1, 0]$
 - $o : [0, 0, 0, 1]$



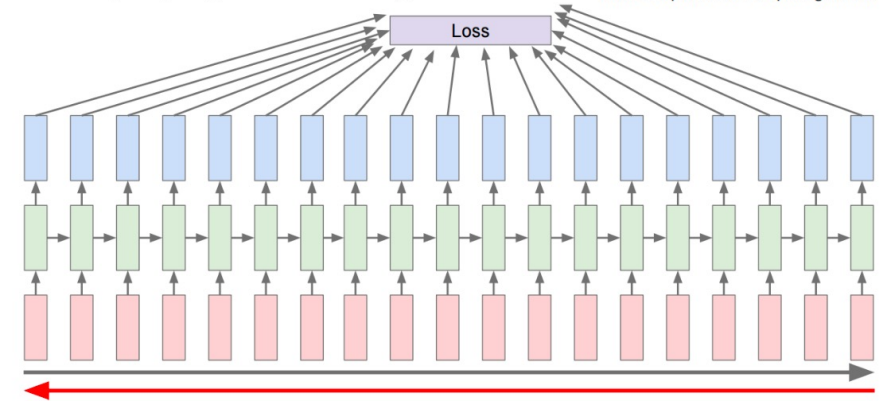
RNN 사용 예시



Language Model

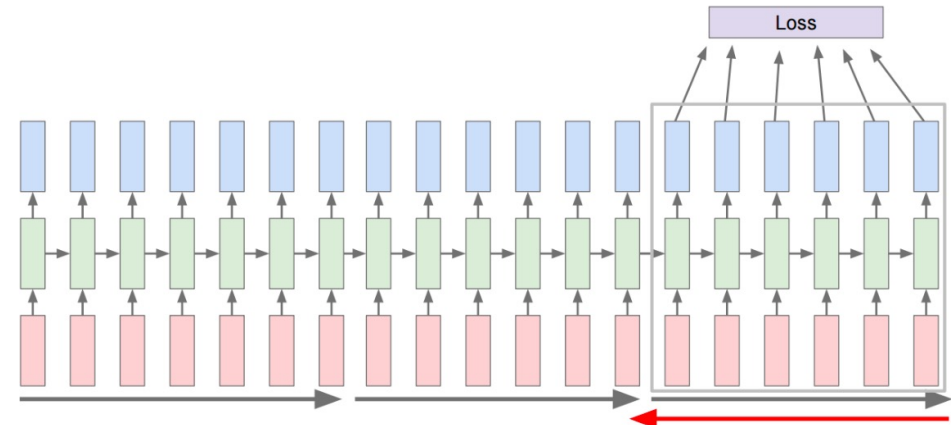
- 매번 존재하는 출력값의 loss로 final loss 계산
“Backpropagation Through Time”
- 시퀀스가 길면? 다 끝나야 한 번 gradient 계산
- 그래서 실제로는 일정 단위로 스텝 잘라서 계산
“Truncated Backpropagation Through Time”

Backpropagation through time



Forward through entire sequence to compute loss, then backward through entire sequence to compute gradient

Truncated Backpropagation through time



RNN의 단점



장기 의존성 Long-Term Dependency

- 데이터가 너무 길면, 뒤로 갈수록 앞쪽 데이터의 입력을 까먹음
- backward pass에서 gradient를 계산할 때 점점 작아짐 “Gradient Vanishing”
- 이걸 해결한 것이 LSTM

