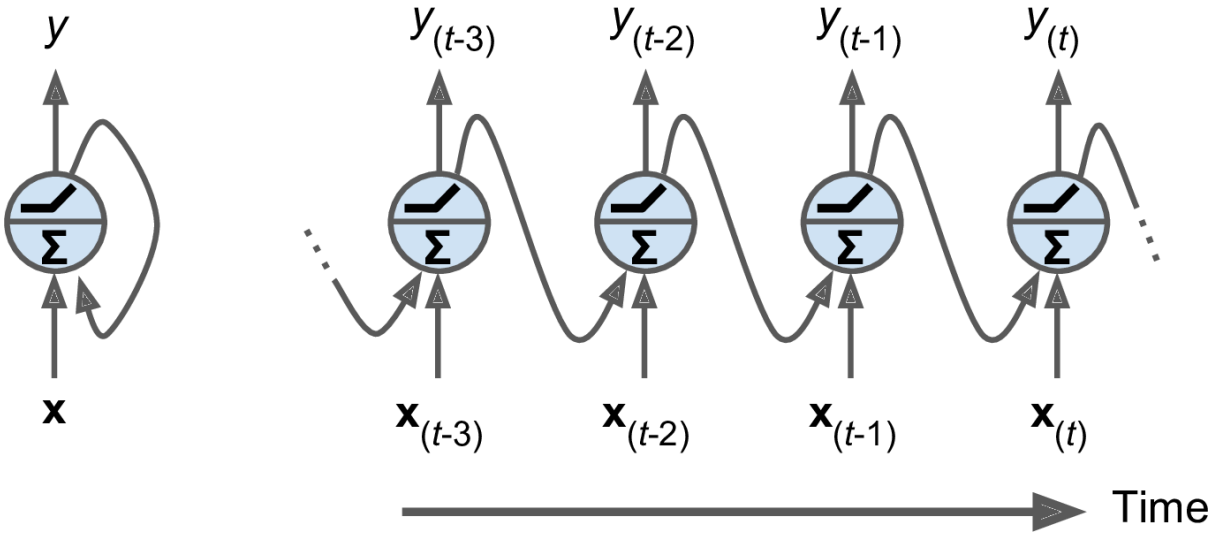
CHAP15. Processing Sequences Using RNNs and CNN (1)

1. 이번 챕터의 목표

- fundamental concepts of RNN (like BTTS)
- forecasting a time series
- how to handle two main difficulties that RNN face
  - Unstable gradient (→ recurrent dropout, recurrent layer normalization)
  - short-term memory (→ LSTM, GRU cells)
    - other neural networks for sequential data : regular dense network, CNN(e.g WaveNet)

2. Recurrent Neurons and Layers

가. 순환신경망의 개념



순환신경망의 입출력을 시간에 따라 펼친 그림

- Simple RNN은 output을 출력하고 다음 스텝의 자신에게 출력을 input으로 보내는 뉴런 하나로 구성
- x(t)는 이전 time step의 출력값인 y(t-1)을 입력값으로 받게 됨
- 그렇다면 t 시점의 입력값은 x(t)와 y(t-1)의 2가지가 되고, t=0인 최초 시점의 경우 y(t-1)이 존재하지 않기 때문에 0의 값을 받게 됨
- 순환 뉴런은 x(t)와 y(t-1)의 각 입력값을 위한 2개의 가중치 벡터를 갖게 됨
- RNN 출력 벡터 계산식

$$Y(t) = \phi(X(t)Wx + Y(t-1)Wy + b) = \phi([X(t)Y(t-1)]W + b)$$

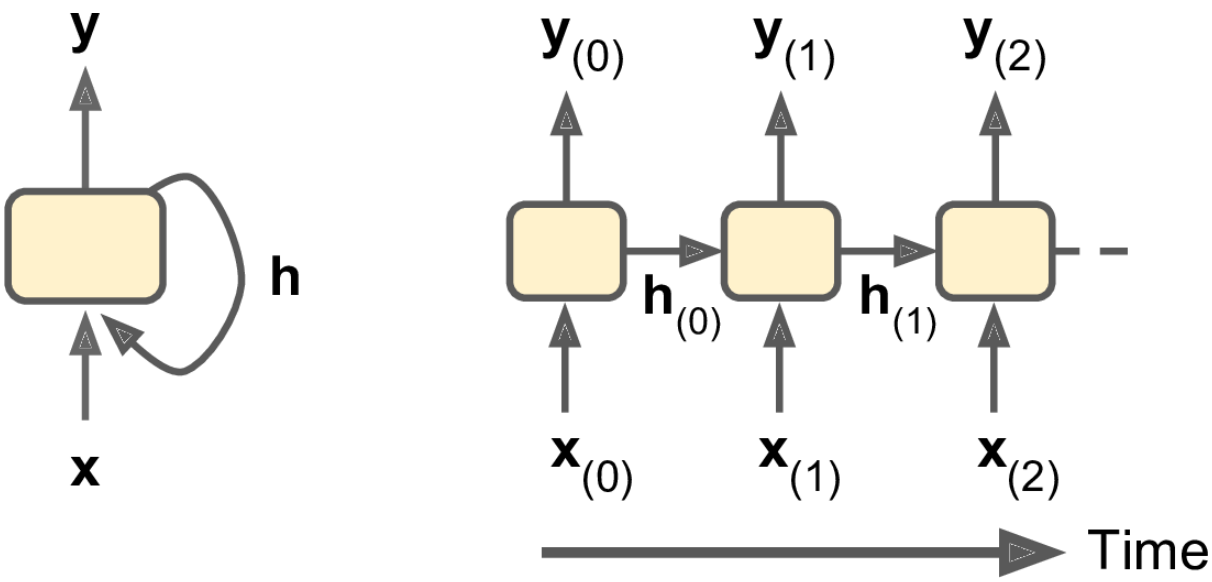
$$W = [WxWy]$$

나. Memory Cells

- y(t)는 이전 타임스텝에서 계산되었던 모든 input의 출력 결과이므로 memory 형태라고 생각할 수 있음
- 타임스텝에 따른 상태를 보존하는 신경망을 memory cell (또는 cell) 이라고 부름
- memory cell은 RNN의 layer로 치환해서 생각하면 됨 (= RNN에서의 Layer를 cell로 부른다)
- cell's state는 해당 타임스텝의 입력 x(t)와 이전 타임 스텝의 상태 h(t-1)를 의미

$$h(t) = f(h(t-1), x(t))$$

- Simple RNN에서 cell's state는 출력값인 y(t)와 동일하지만, 복잡한 셀에서는 y(t)와 h(t)가 같지 않음 (아래 그림 참조)



Input and Output Sequence

RNN의 입출력 시퀀스 종류

1. Sequence-to-sequence network

: N개의 시퀀셜 데이터를 입력하면 N개의 예측 출력값을 가져옴

e.g) 오늘을 포함해 최근 N일치의 주식가격을 주입하면 N-1일 전부터 내일까지의 예측값 출력

2. Sequence-to-vector network

: 시퀀셜 데이터를 입력하면 하나의 벡터를 출력함

e.g) 영화리뷰(연속된 단어 데이터)를 입력하면 -1부터 1까지의 감성 점수 출력

3. vector-to-sequence network

: 하나의 벡터를 입력하고, 시퀀셜 데이터를 출력함

e.g) 이미지를 입력하면 이미지에 대한 캡션(연속된 단어 데이터)를 출력

4. Encoder(sequence-to-vector) - Decoder(vector-to-sequence)

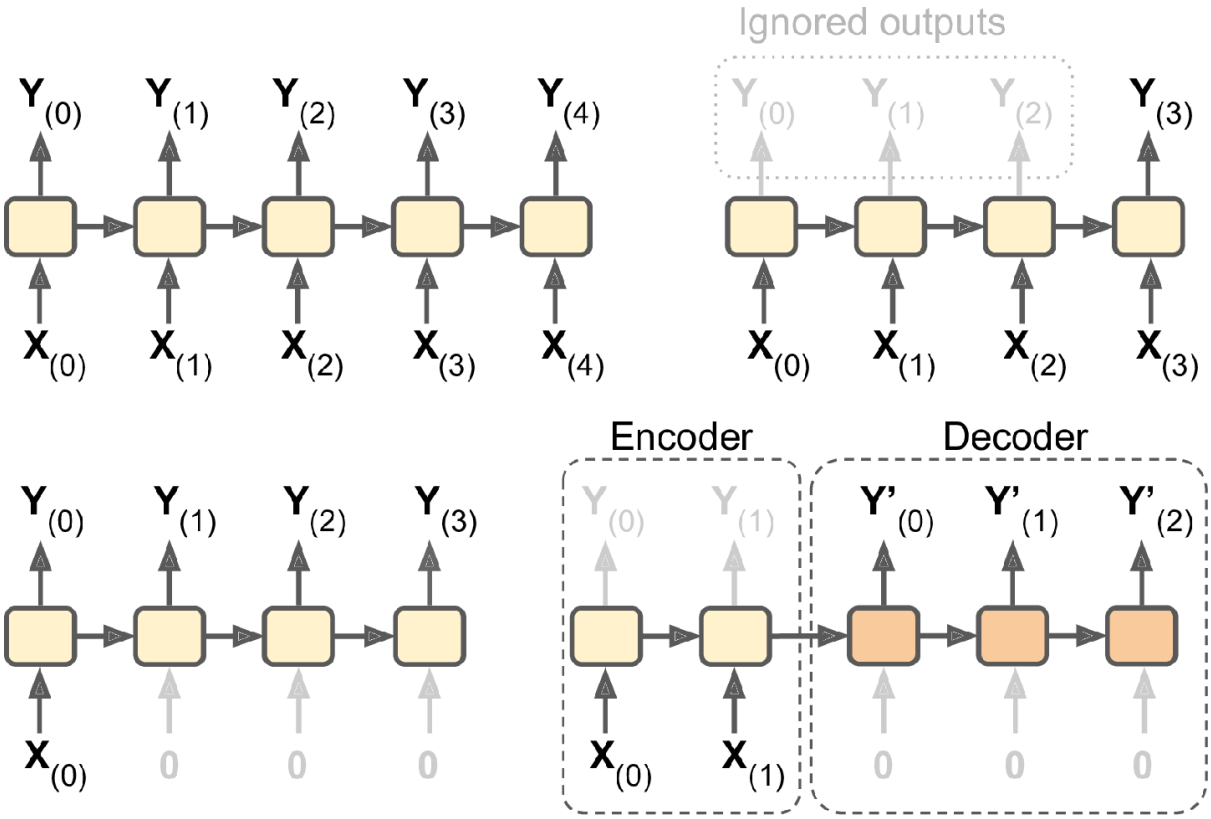
: - Sequential data를 벡터로 바꾼 뒤 , 그 벡터를 다시 Sequential data로 출력하는 방법

- 해당 네트워크는 Sequence-to-Sequence 모델보다 좋은 성능을 보임

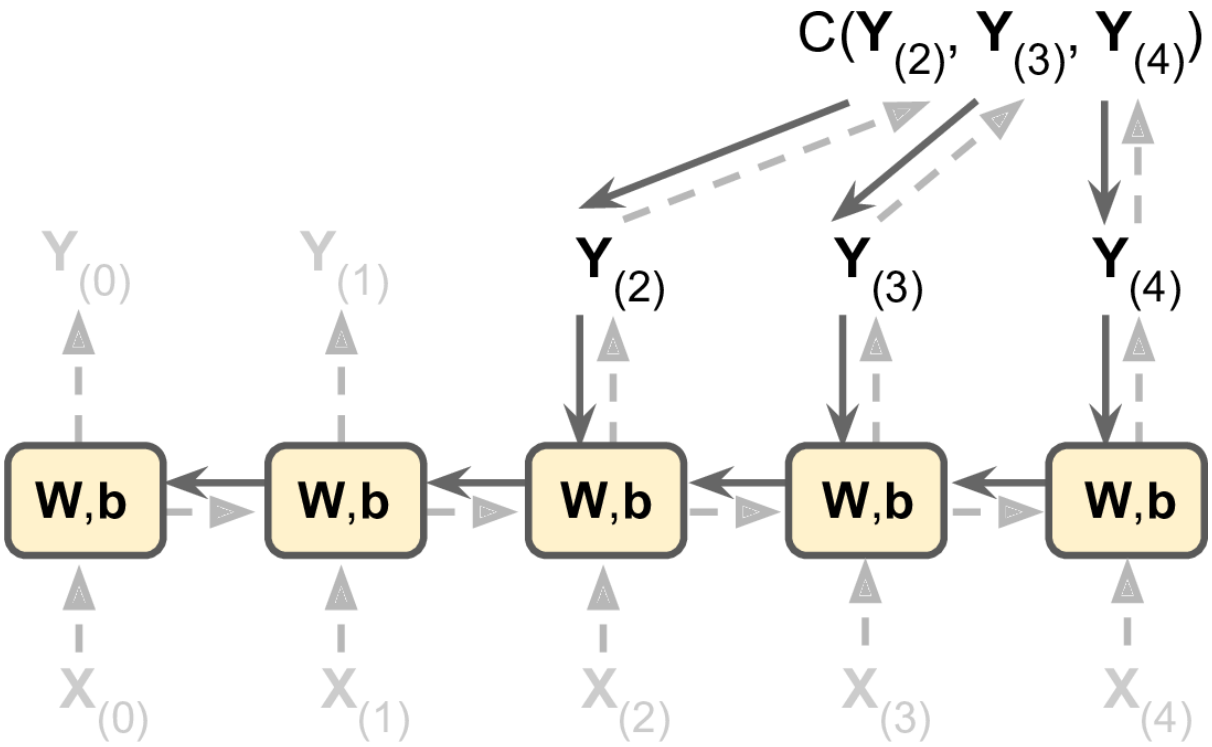
e.g) 한 언어의 문장을 다른 언어의 문장으로 번역

문장의 마지막 단어가 번역의 첫번째 단어에 영향을 주기 때문에 sequence to sequence보다 좋은성능을 보인다?

→ sequence-to-sequence에서는 T번째를 예측하기 위해선 T-1까지의 데이터만 입력 값으로 받아서 활용하게 됨. sequence-to-vector에서 vector-to-sequence를 이용하면, Sequence data를 벡터로 만들기 위해 모든 데이터가 전부 입력될 때까지 기다려야 한다. 그래서 Sequnce에서의 T번째를 예측하기 위해 T번째 이후의 데이터도 모두 고려되기 때문에 더 정확한 예측(번역)을 할 수 있다는 이야기로 보임



Training RNN



- Time step에 따라 순환신경망을 펼치고 다른 신경망과 마찬가지로 Backpropagation을 사용
- 이렇게 Time step대로 신경망을 펼쳐서 역전파를 이용하는 방법을 Backpropagation through time(BPTT)라고 부름

  (1) 순방향으로 입력값이 네트워크를 통과(점선 표현)

  (2) 출력값(벡터 혹은 시퀀스)을 기반으로 비용함수를 평가하고, 역방향으로 그래디언트를 계산해가면서 가중치를 업데이트함(실선 표현)

  위의 그림 예시에서는 Y(2),Y(3),Y(4)의 세 개 출력을 기반으로 비용함수가 계산되었고, RNN은 하나의 가중치 벡터 W를 사용하기 때문에 세 개 출력값을 기반으로 전체 타임스텝의 벡터가 수정되게 됨.