# 인 공 지 능

## [순환 신경망 I]

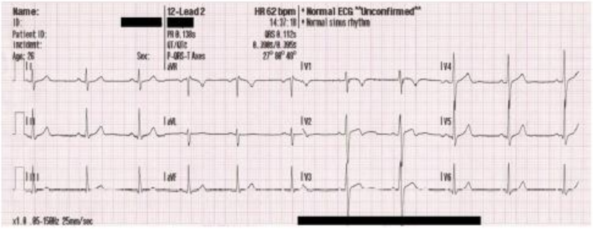


jaekoo@kookmin.ac.kr

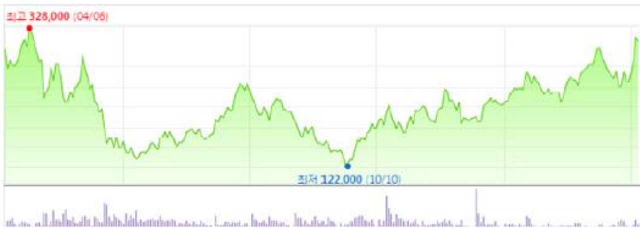
# 미리보기

## 시간성time series 데이터

- 특징이 순서를 가지므로 순차 데이터sequential data라 부름
  (지금까지 다룬 데이터는 어느 한 순간에 취득한 정적인 데이터이고 고정 길이임fixed input)
- 순차 데이터는 동적이며 보통 가변 길이임variable-length input



(a) 심전도 신호




(b) 주식 시세

(c) 음성 신호

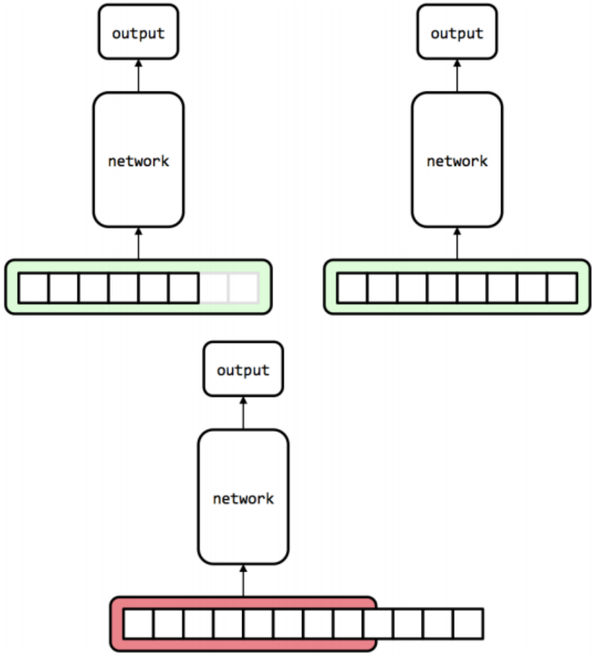
내려갈 때 보았네 올라갈 때 보지 못한 그 꽃

(d) 문장

ATGCTTCGGCAAGACTCAAAAAATA

(e) 유전자 열

**그림 8-1 순차 데이터**




고정 길이 입력과 가변 길이 입력

# 미리보기

- 순환 신경망recurrent neural networks과 LSTM
  - 순환 신경망은 시간성 정보를 활용하여 순차 데이터를 처리하는 효과적인 학습 모델
  - 매우 긴 순차 데이터 (예, 30단어 이상의 긴 문장)를 처리에는

    장기 의존성long-term dependency을 잘 다루는 LSTM을 주로 사용 (LSTM은 선별 기억 능력을 가짐)

- 최근에는 순환 신경망도 생성 모델로 사용
  - 예, CNN과 LSTM이 협력하여 자연 영상에 주석 생성하는 문제를 해결 (8.5.3절)

# 8.1 순차 데이터

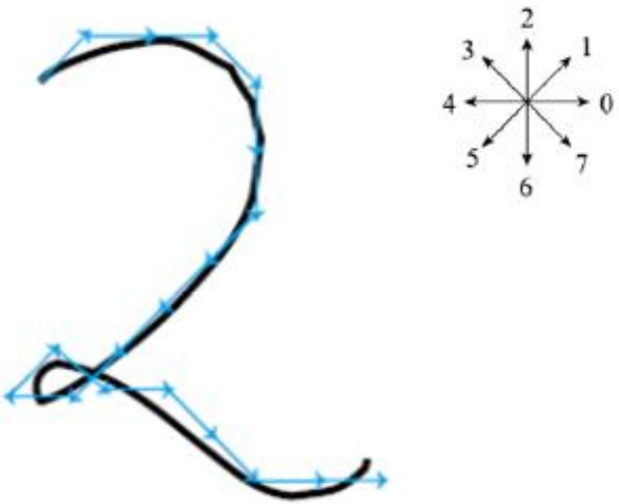
- 8.1.1 순차 데이터의 표현
- 8.1.2 순차 데이터의 특성

- 많은 응용
  - 심전도 신호를 분석하여 심장 이상 유무 판정
  - 주식 시세 분석하여 사고 파는 시점 결정
  - 음성 인식을 통한 지능적인 인터페이스 구축
  - 기계 번역기 또는 자동 응답 장치 제작
  - 유전자 열 분석을 통한 치료 계획 수립 등

# 8.1.1 순차 데이터의 표현

## 순차 데이터의 예시

- 온라인 숫자와 3채널 심전도 신호




x=10076655541707700

(a) 온라인 숫자

(b) 심전도 신호(3채널)

**그림 8-2 순차 데이터의 표현**

# 8.1.1 순차 데이터의 표현

## ■ 순차 데이터의 일반적 표기

- 벡터의 벡터 (벡터의 요소가 벡터)

$$\mathbf{x} = \left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(T)}\right)^{\mathrm{T}} \tag{8.1}$$

- 온라인 숫자의 요소는 1차원, 심전도의 요소는 3차원
  - 예, 심전도 신호 (초당 100번 샘플링하고 2분간 측정한다면 길이는 $T$=12000)

$$\mathbf{x} = \left(\begin{pmatrix} 0.3 \\ 0.1 \\ 0.2 \end{pmatrix}, \begin{pmatrix} 0.5 \\ 0.6 \\ 0.4 \end{pmatrix}, \cdots \cdots \right)^{\mathrm{T}}$$

## ■ 훈련집합 $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, $\mathbb{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n\}$

- 각 샘플은 식 (8.2)로 표현

$$\mathbf{x} = \left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(T)}\right)^{\mathrm{T}}, \quad \mathbf{y} = \left(\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \cdots, \mathbf{y}^{(L)}\right)^{\mathrm{T}} \tag{8.2}$$

# 8.1.1 순차 데이터의 표현

- 대표적인 순차 데이터인 문자열의 표현
  - 예, 기계 번역에서

    입력 $\mathbf{x}$가 “April is the cruelest month.”이고

    출력 $\mathbf{y}$가 “사월은 가장 잔인한 달”일 때, 식 (8.2) 표기법으로 어떻게 표현할까?

- 사전dictionary or term을 사용하여 표현
  - 사전 구축 방법
    - 사람이 사용하는 단어를 모아 구축

      또는 주어진 말뭉치를 분석하여 단어를 자동 추출하여 구축
      - 예, 영어를 불어로 번역하는 논문 [Cho2014b]에서는 사용 빈도가 가장 높은 3만 개 단어로 사전 구축함

  - 사전을 사용한 텍스트 순차 데이터의 표현 방법
    - 단어가방BoW (bag of words)
    - 원핫 코드one-hot code
    - 단어 임베딩word embedding

# 8.1.1 순차 데이터의 표현

## 단어 가방

- 단어 각각의 빈도수를 세어 $m$차원의 벡터로 표현 ($m$은 사전 크기)
- 한계
  - 정보 검색에 주로 사용되지만, 기계 학습에는 부적절
    - (“April is the cruelest month”와 “The cruelest month is April”은 같은 특징 벡터로 표현되어 시간성 정보가 사라짐)

**Document 1**

The quick brown fox jumped over the lazy dog's back.

**Document 2**

Now is the time for all good men to come to the aid of their party.

| Term | Document 1 | Document 2 |
|---|---|---|
| aid | 0 | 1 |
| all | 0 | 1 |
| back | 1 | 0 |
| brown | 1 | 0 |
| come | 0 | 1 |
| dog | 1 | 0 |
| fox | 1 | 0 |
| good | 0 | 1 |
| jump | 1 | 0 |
| lazy | 1 | 0 |
| men | 0 | 1 |
| now | 0 | 1 |
| over | 1 | 0 |
| party | 0 | 1 |
| quick | 1 | 0 |
| their | 0 | 1 |
| time | 0 | 1 |

## 원핫 코드

- 해당 단어의 위치만 1로 표시
  - 예, “April is the cruelest month”는 $\mathbf{x} = \left((0,0,1,0,0,0,\cdots)^{\mathrm{T}}, (0,0,0,0,1,0,\cdots)^{\mathrm{T}},\cdots\right)^{\mathrm{T}}$로 표현

    ← $m$차원 벡터를 요소로 가진 5차원 벡터
- 한계
  - 한 단어를 표현하는데 $m$차원 벡터를 사용하는 비효율
  - 단어 간의 유사성을 측정할 수 없음

Vocabulary:
Man, woman, boy, girl, prince, princess, queen, king, monarch

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| man | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| woman | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| boy | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| girl | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| prince | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| princess | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| queen | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| king | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| monarch | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

# 8.1.1 순차 데이터의 표현

## 단어 임베딩

- 단어 사이의 상호작용을 분석하여 새로운 공간으로 변환 (보통 $m$보다 훨씬 낮은 차원으로 변환)
- 변환 과정은 학습이 말뭉치를 훈련집합으로 사용하여 알아냄
- 예, word2vec [Cho2014b]는 $m$=30000차원을  620차원으로 변환

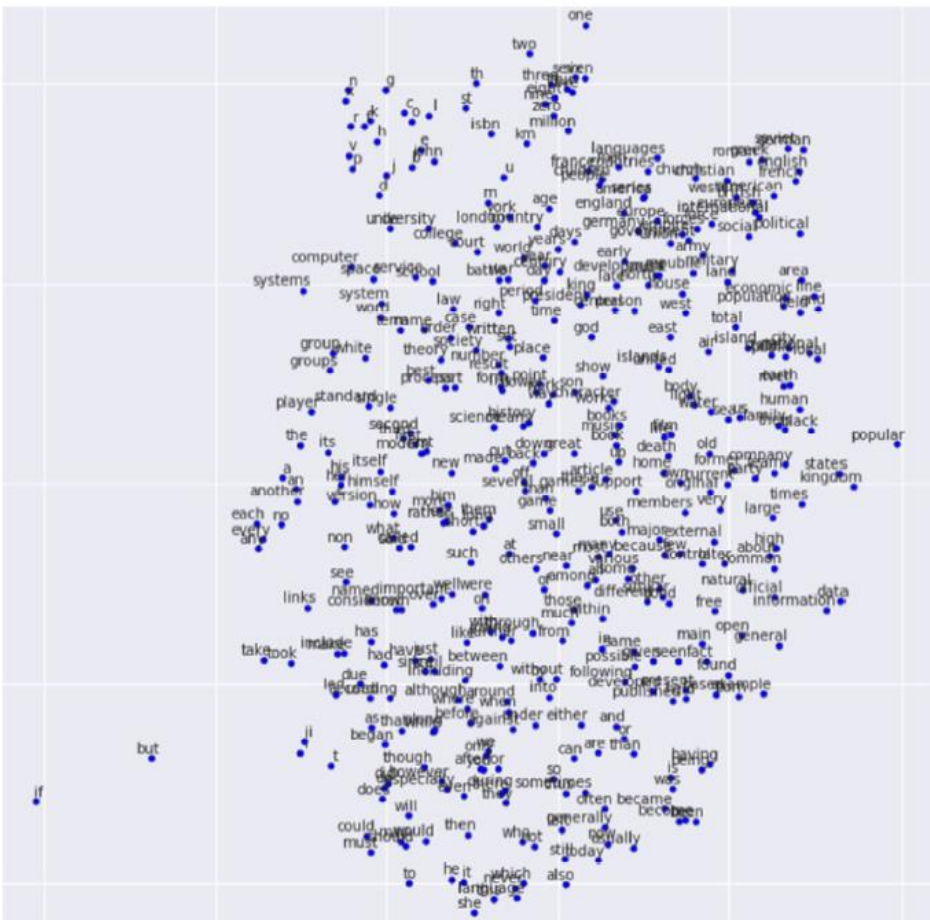
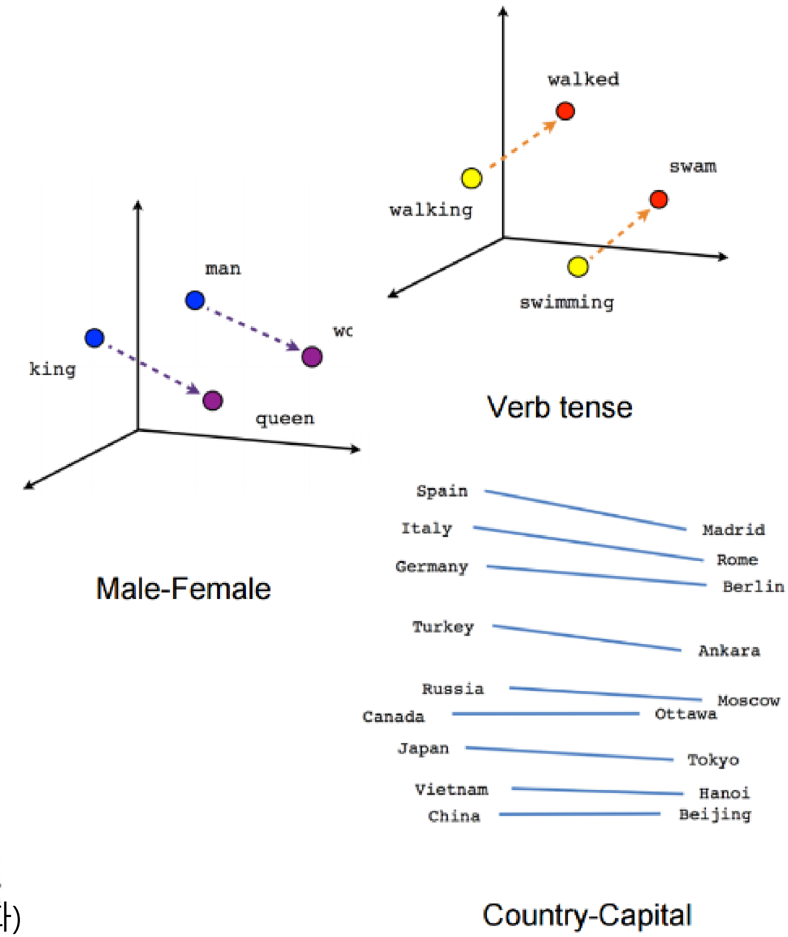


그림 8-3 단어 임베딩 (학습에 의해 임베딩된 단어들을 t-SNE로 시각화)

학습된 단어 사이의 상호 작용의 예



# 8.1.2 순차 데이터의 특성

- 특징이 나타나는 순서가 중요
  - “아버지가 방에 들어가신다.”를 “아버지 가방에 들어가신다.”로 바꾸면 의미가 크게 훼손
  - 비순차 데이터에서는 순서를 바꾸어도 무방
- 샘플마다 길이가 다름
  - [그림 8-2]의 예제
  - 순환 신경망은 은닉층에 순환 연결을 부여하여 가변 길이 수용
- 문맥 의존성
  - 비순차 데이터는 공분산이 특징 사이의 의존성을 나타냄
  - 순차 데이터에서는 공분산은 의미가 없고, 대신 문맥 의존성이 중요함
    - 예, “그녀는 점심때가 다 되어서야 …. 점심을 먹었는데, 철수는 …”에서 “그녀는”과 “먹었는데”는 강한 문맥 의존성을 가짐
    - 특히 이 경우 둘 사이의 간격이 크므로 장기 의존성이라 부름 ← LSTM으로 처리

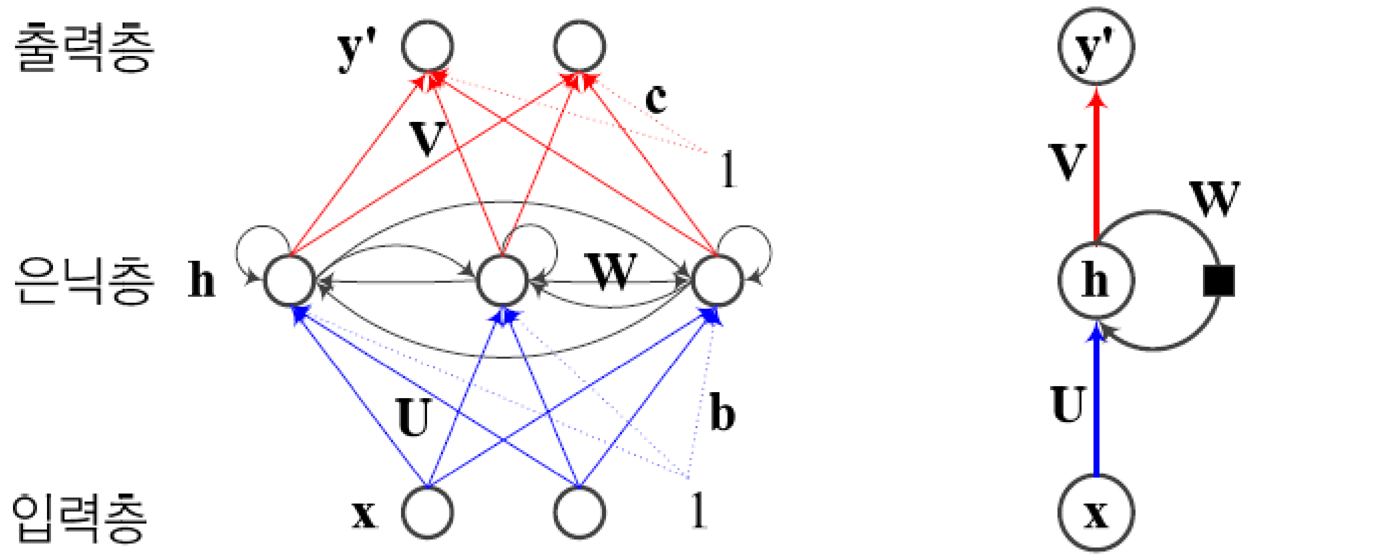
# 8.2 순환 신경망RNN (recurrent neural network )

- 8.2.1 구조
- 8.2.2 동작
- 8.2.3 BPTT 학습
- 8.2.4 양방향 RNN

- 순환 신경망(RNN)이 갖추어야 할 세 가지 필수 기능
  - 시간성: 특징을 순서대로 한 번에 하나씩 입력해야 한다.
  - 가변 길이: 길이가 $T$인 샘플을 처리하려면 은닉층이 $T$번 나타나야 한다. $T$는 가변적이다.
  - 문맥 의존성: 이전 특징 내용을 기억하고 있다가 적절한 순간에 활용해야 한다.

# 8.2.1 구조

- RNN의 구조
  - 기존 깊은 신경망과 유사
    - 입력층, 은닉층, 출력층을 가짐
  - 다른 점은 은닉층이 순환 연결recurrent edge (recurrent connection)를 가진다는 점
    - 시간성, 가변 길이, 문맥 의존성을 모두 처리할 수 있음
    - 순환 연결은 $t$-1 순간에 발생한 정보를 $t$ 순간으로 전달하는 역할



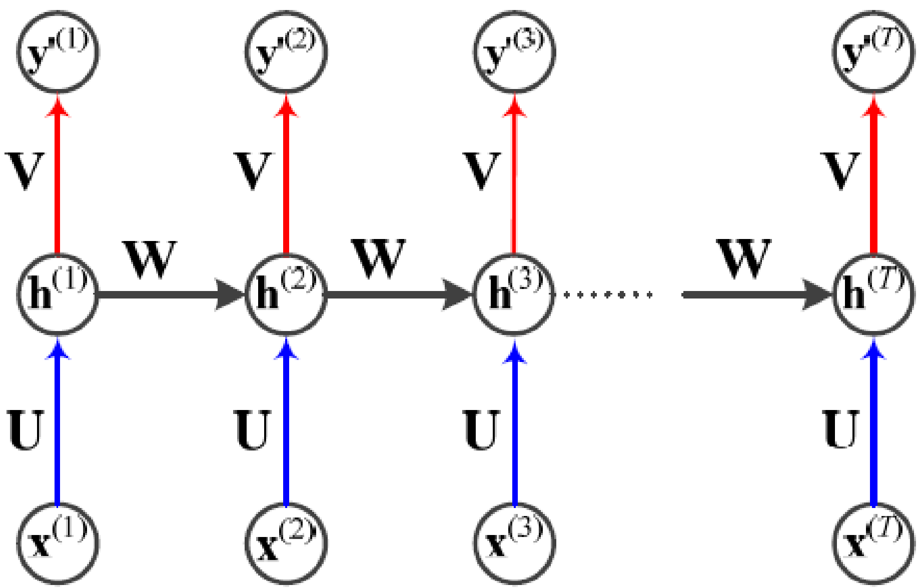
(a) RNN (b) 축약형으로 그림

# 8.2.1 구조

- 수식으로 쓰면,

$$\mathbf{h}^{(t)} = f\left(\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}; \Theta\right) \tag{8.3}$$

  - t=1 순간에 계산하고, 그 결과를 가지고 t=2 순간에 계산하고, 그 결과를 가지고 t=3 순간에 계산하고, …, $T$ 순간까지 반복
  - 일반적으로 $t$ 순간에는 $t$-1 순간의 은닉층 값 (상태) $\mathbf{h}^{(t-1)}$과 $t$ 순간의 입력 $\mathbf{x}^{(t)}$를 받아 $\mathbf{h}^{(t)}$로 전환함
  - Θ는 순환 신경망의 매개변수




(c) 펼침

그림 8-4 RNN의 구조

# 8.2.1 구조

- 펼쳐서 다시 그리면,

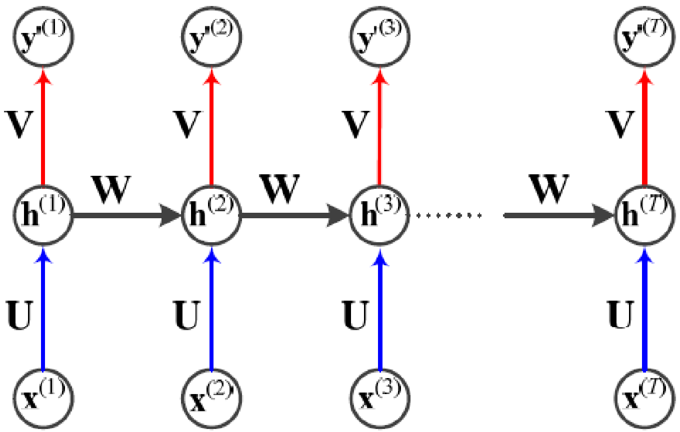



**그림 8-4** RNN의 구조

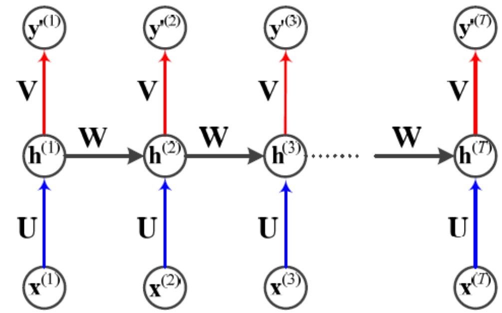
- 식 (8.3)을 펼치면,

$$
\begin{aligned}
\mathbf{h}^{(T)} &= f(\mathbf{h}^{(T-1)}, \mathbf{x}^{(T)}; \Theta) \\
&= f(f(\mathbf{h}^{(T-2)}, \mathbf{x}^{(T-1)}; \Theta), \mathbf{x}^{(T)}; \Theta) \\
&\quad \vdots \\
&= f(f(\cdots f(\mathbf{h}^{(1)}, \mathbf{x}^{(2)}; \Theta), \cdots, \mathbf{x}^{(T-1)}; \Theta), \mathbf{x}^{(T)}; \Theta) \\
&= f(f(\cdots f(f(\mathbf{h}^{(0)}, \mathbf{x}^{(1)}; \Theta), \mathbf{x}^{(2)}; \Theta), \cdots, \mathbf{x}^{(T-1)}; \Theta), \mathbf{x}^{(T)}; \Theta)
\end{aligned}
\tag{8.4}
$$

# 8.2.1 구조

- 순환 신경망의 매개변수 (가중치 집합)는 $\Theta = \{\mathbf{U}, \mathbf{W}, \mathbf{V}, \mathbf{b}, \mathbf{c}\}$
  - $\mathbf{U}$는 입력층과 은닉층을 연결하는 $p$*$d$ 행렬
  - $\mathbf{W}$는 은닉층과 은닉층을 연결하는 $p$*$p$ 행렬
  - $\mathbf{V}$는 은닉층과 출력층을 연결하는 $q$*$p$ 행렬
  - $\mathbf{b}, \mathbf{c}$는 바이어스로서 각각 $p$*1과 $q$*1 행렬

  → RNN 학습이란 훈련집합을 최적의 성능으로 예측하는 $\Theta$ 값을 찾는 일



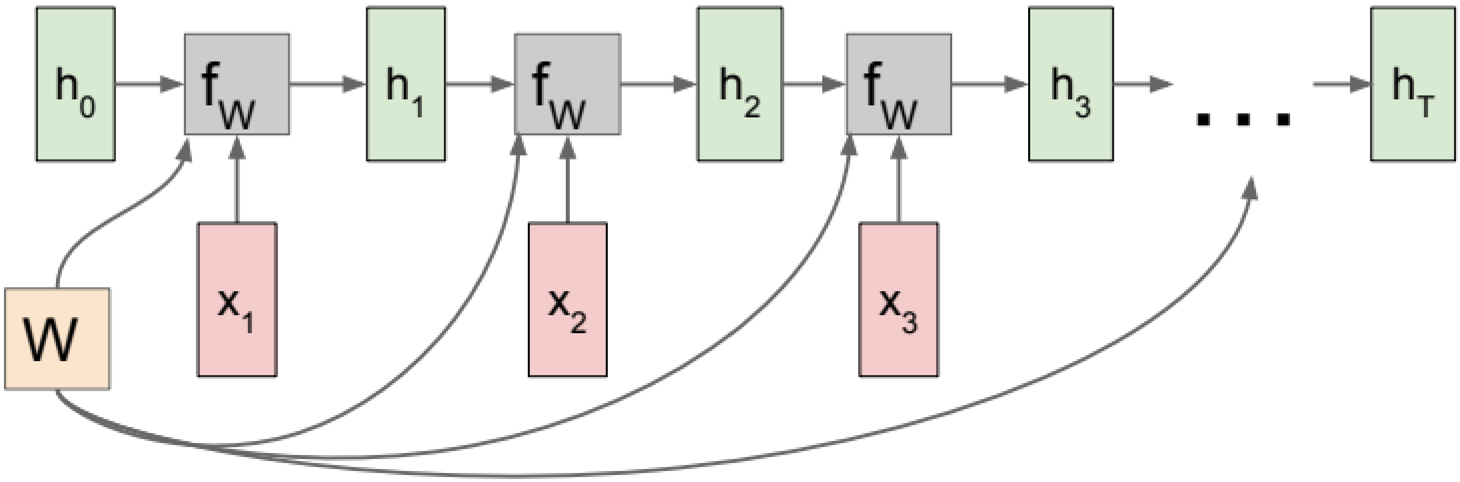

(c) 펼침

그림 8-4 RNN의 구조

- 매개변수 공유
  - 매 순간 다른 값을 사용하지 않고 같은 값을 공유함 ([그림 8-4(c)])
  - 공유의 장점
    - 추정할 매개변수 수가 획기적으로 줄어듦
    - 매개변수의 수가 특징 벡터의 길이 $T$에 무관
    - 특징이 나타나는 순간이 뒤바뀌어도 같거나 유사한 출력을 만들 수 있음
      – 예, “어제 이 책을 샀다”와 “이 책을 어제 샀다”를 비슷한 영어 문장으로 번역할 수 있음

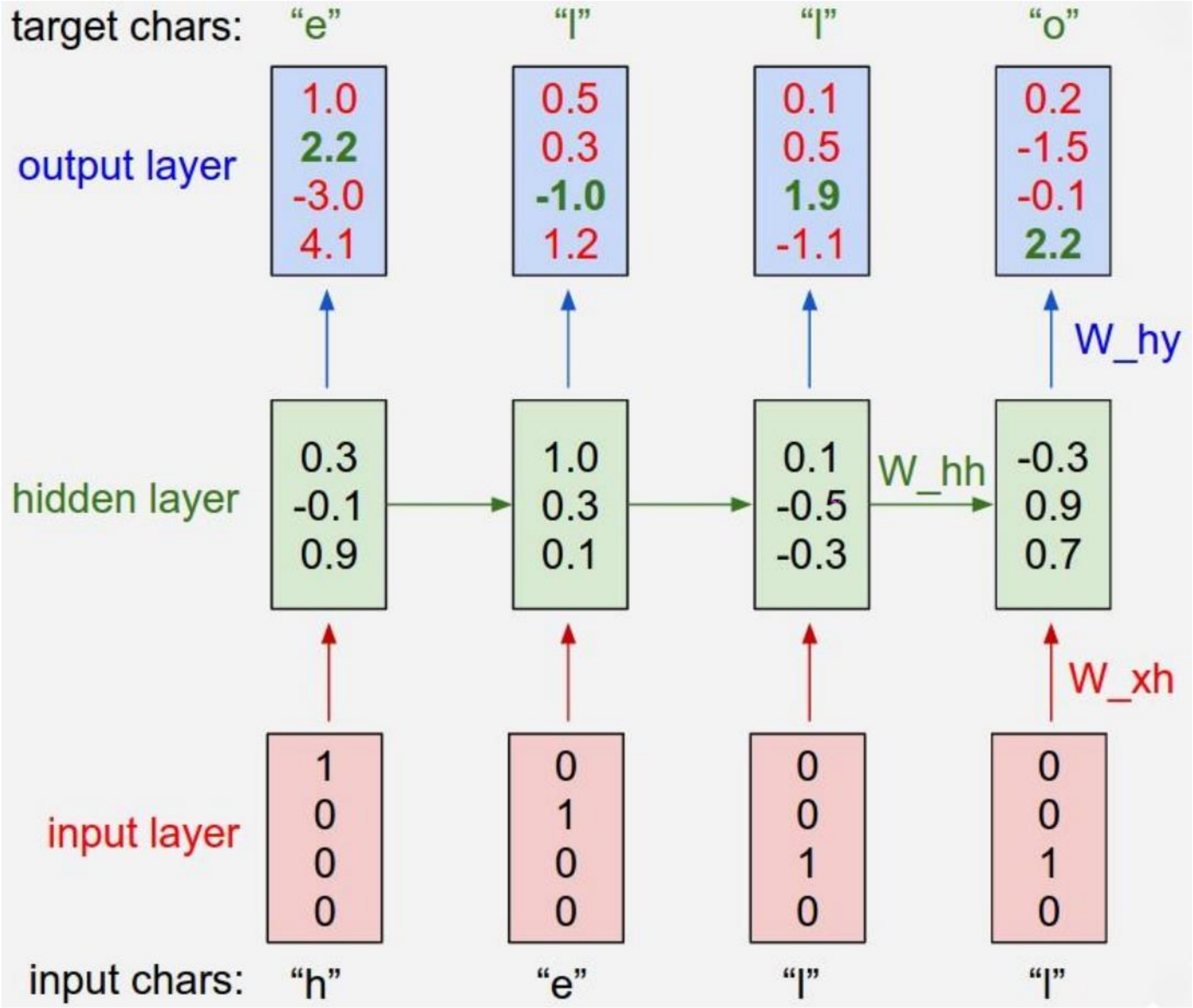
# 8.2.1 구조

- 기본 RNN의 연산 그래프
  - 같은 가중치 행렬 W을 매 시간마다 재사용함



# 8.2.1 구조

- 문자 단위 언어 모델의 예



# 8.2.1 구조

## 다양한 RNN 구조

- [그림 8-4]는 입력의 개수 $T$와 출력의 개수 $L$이 같은 경우
- [그림 8-5]는 $T \neq L$인 경우
  - 왼쪽은 퀴즈풀이 응용 예 $(L = 1)$, 입력은 “인공지능 담당 교수는?”, 출력은 “이재구”

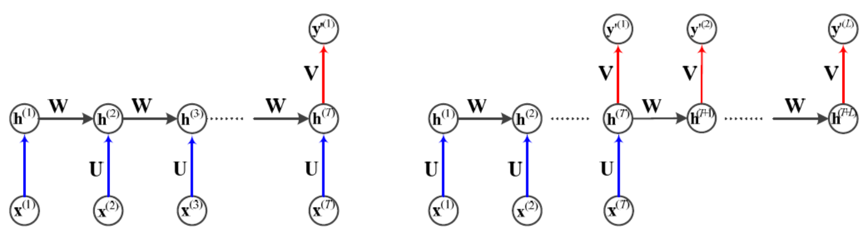
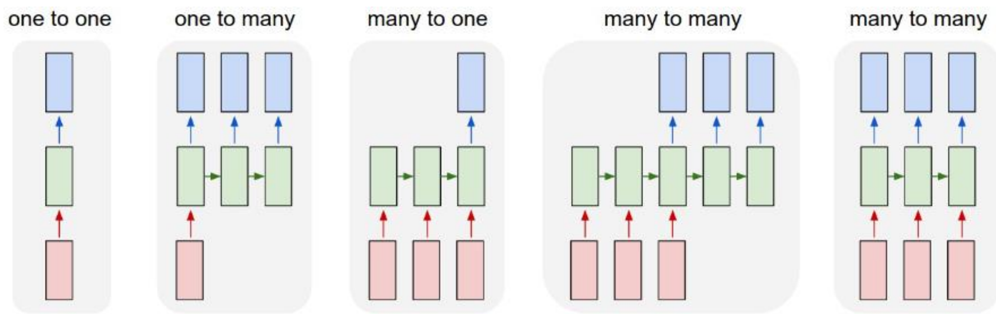
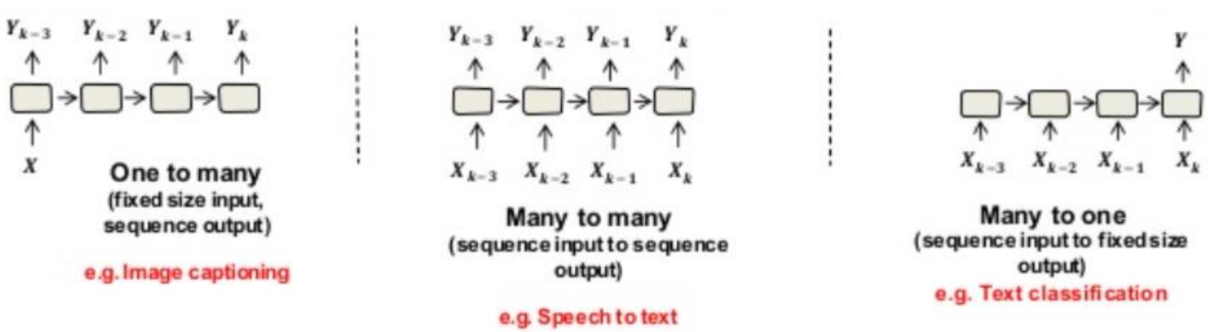


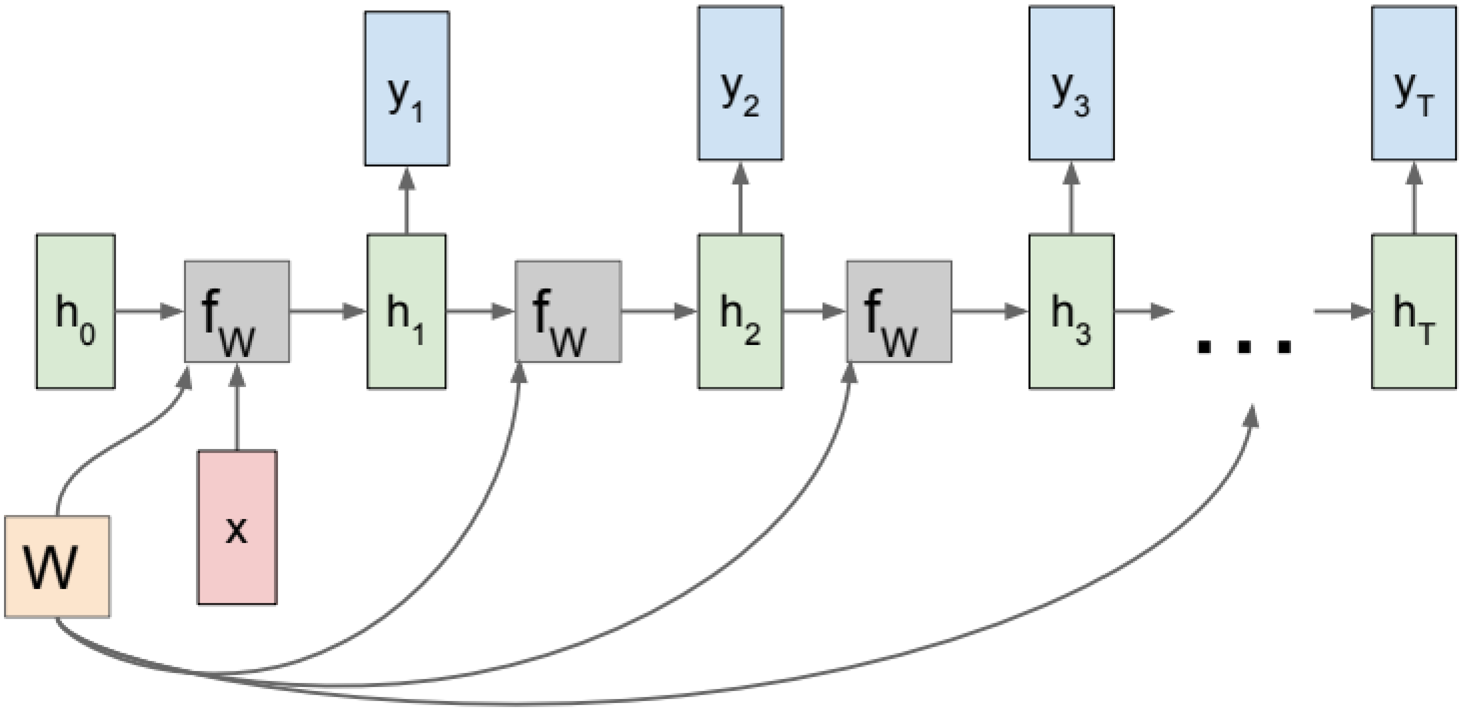
그림 8-5 RNN의 여러 가지 변형



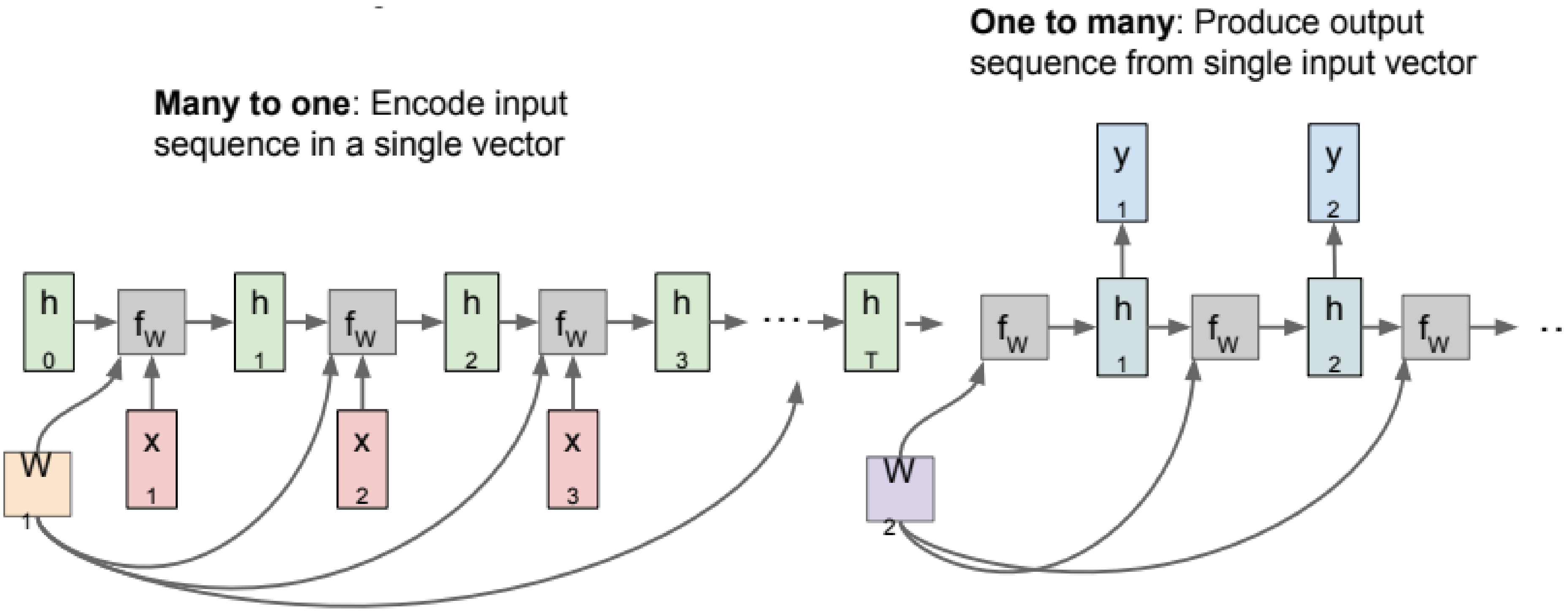
# 8.2.1 구조

- 일대다one to many RNN의 연산 그래프



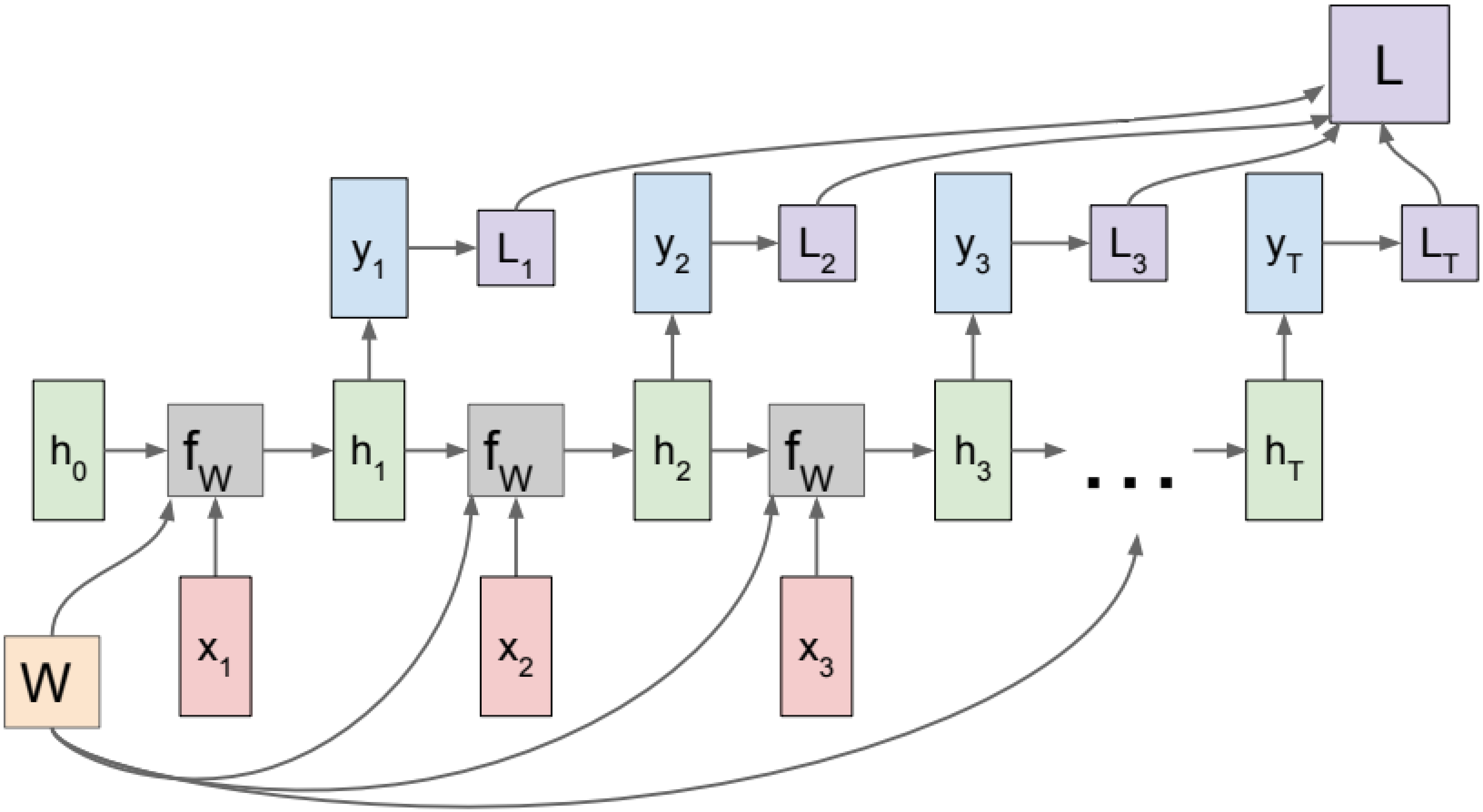
# 8.2.1 구조

- 문장 대 문장sequence to sequence RNN의 연산 그래프
  - 다대일과 일대다의 조합



# 8.2.1 구조

- 다대다many to many RNN의 연산 그래프



# 8.2.2 동작

## ■ RNN의 가중치

$$\mathbf{U}=\begin{array}{cl}\begin{pmatrix} u_{11} & u_{12} & \cdots & u_{1d} \\ u_{21} & u_{22} & \cdots & u_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ u_{p1} & u_{p2} & \cdots & u_{pd} \end{pmatrix} & \begin{array}{l} \\ \leftarrow \mathbf{u}_2 \\ \\ \\ \end{array}\end{array}, \mathbf{V}=\begin{pmatrix} v_{11} & v_{12} & \cdots & v_{1p} \\ v_{21} & v_{22} & \cdots & v_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ v_{q1} & v_{q2} & \cdots & v_{qp} \end{pmatrix}, \mathbf{W}=\begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1p} \\ w_{21} & w_{22} & \cdots & w_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ w_{p1} & w_{p2} & \cdots & w_{pp} \end{pmatrix} \qquad (8.5)$$

- $\mathbf{u}_j=(u_{j1},u_{j2},\cdots,u_{jd})$는 $\mathbf{U}$ 행렬의 $j$번째 행 ($h_j$에 연결된 선의 가중치들)

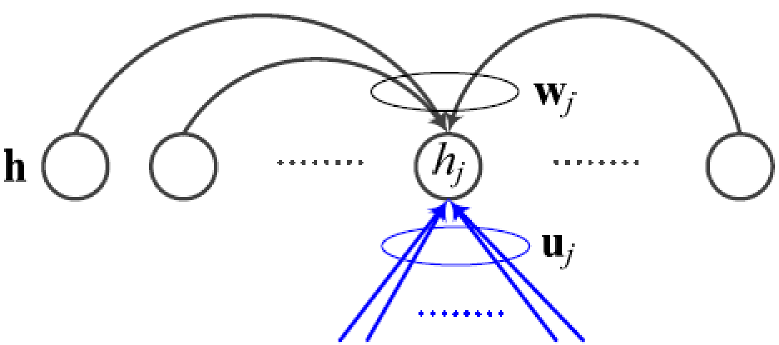



그림 8-6 은닉 노드의 가중치 표기

# 8.2.2 동작

- 은닉층의 계산

$$h_j^{(t)} = \tau\left(a_j^{(t)}\right), \qquad j = 1,2,\cdots,p$$
$$\text{이때, } a_j^{(t)} = \mathbf{w}_j\mathbf{h}^{(t-1)} + \mathbf{u}_j\mathbf{x}^{(t)} + b_j \tag{8.6}$$

  - MLP와 유사 ($\mathbf{w}_j\mathbf{h}^{(t-1)}$ 항을 제외하면 MLP와 동일함)
  - 행렬 표기로 쓰면,

$$\mathbf{h}^{(t)} = \tau(\mathbf{a}^{(t)})$$
$$\text{이때, } \mathbf{a}^{(t)} = \mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{U}\mathbf{x}^{(t)} + \mathbf{b} \tag{8.7}$$

- 은닉층 계산이 끝난 후 출력층의 계산

$$\mathbf{o}^{(t)} = \mathbf{V}\mathbf{h}^{(t)} + \mathbf{c} \tag{8.8}$$

$$\mathbf{y}'^{(t)} = \text{softmax}(\mathbf{o}^{(t)}) \tag{8.9}$$

# 8.2.2 동작

**예제 8-1** **RNN의 동작**

[그림 8-7]은 간단한 예제 RNN이다. 그림을 간결하게 하려고 가중치가 0인 에지는 숫자를 기입하지 않았다. 식 (8.5)에 따라 이 RNN의 매개변숫값은 다음과 같다.

$$\mathbf{U}=\begin{pmatrix}0.1 & 0.1\\ 0.0 & 0.0\\ 0.0 & -0.1\end{pmatrix},\ \mathbf{W}=\begin{pmatrix}0.1 & 0.1 & 0.0\\ 0.0 & 0.0 & 0.0\\ 0.2 & -0.1 & -0.1\end{pmatrix},\ \mathbf{V}=\begin{pmatrix}0.0 & 0.1 & 0.0\\ -0.2 & 0.0 & 0.0\end{pmatrix},\ \mathbf{b}=\begin{pmatrix}0.0\\ 0.0\\ 0.2\end{pmatrix},\ \mathbf{c}=\begin{pmatrix}0.2\\ 0.1\end{pmatrix}$$

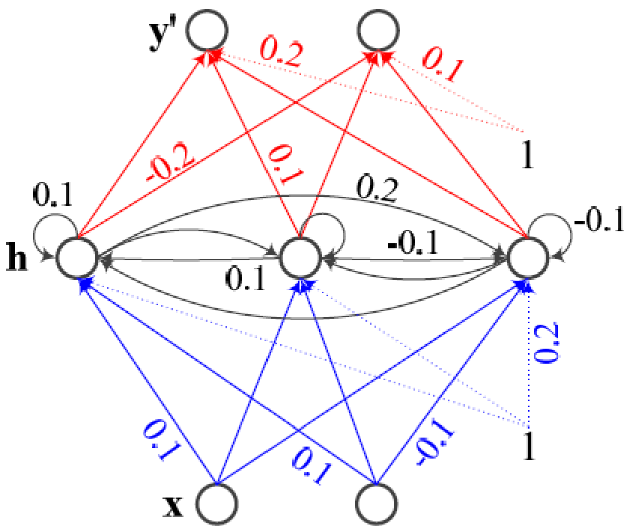
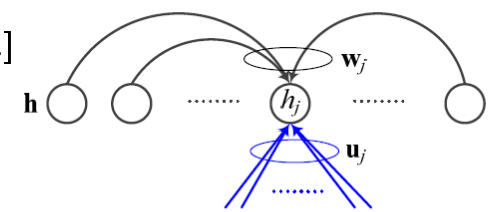
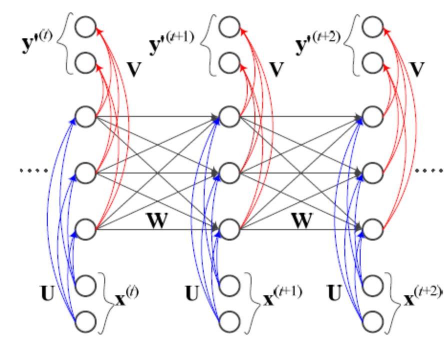



그림 8-7 예제 RNN

[참고]




이 RNN에 샘플 $\mathbf{x}=\left(\begin{pmatrix}0.0\\1.0\end{pmatrix},\begin{pmatrix}0.0\\0.1\end{pmatrix},\begin{pmatrix}0.1\\-0.2\end{pmatrix},\begin{pmatrix}0.5\\0.0\end{pmatrix}\right)^{\mathrm{T}}$, $\mathbf{y}=\left(\begin{pmatrix}0.7\\0.4\end{pmatrix},\begin{pmatrix}0.3\\0.4\end{pmatrix},\begin{pmatrix}0.2\\0.4\end{pmatrix},\begin{pmatrix}0.4\\0.5\end{pmatrix}\right)^{\mathrm{T}}$가 주어졌다고 가정하면 다음과 같은 연산이 일어난다.

# 8.2.2 동작

$t$=1일 때, 식 (8.7)과 식 (8.8)에 값을 대입하면 다음과 같다. 활성함수로 tanh를 사용한다고 가정하였다. 은닉층의 초깃값 $\mathbf{h}^{(0)} = (0\ 0\ 0)^{\mathrm{T}}$라고 가정한다.

$$\mathbf{a}^{(1)} = \mathbf{Wh}^{(0)} + \mathbf{Ux}^{(1)} + \mathbf{b} = \begin{pmatrix} 0.1 & 0.1 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 0.2 & -0.1 & -0.1 \end{pmatrix}\begin{pmatrix} 0.0 \\ 0.0 \\ 0.0 \end{pmatrix} + \begin{pmatrix} 0.1 & 0.1 \\ 0.0 & 0.0 \\ 0.0 & -0.1 \end{pmatrix}\begin{pmatrix} 0.0 \\ 1.0 \end{pmatrix} + \begin{pmatrix} 0.0 \\ 0.0 \\ 0.2 \end{pmatrix} = \begin{pmatrix} 0.1 \\ 0.0 \\ 0.1 \end{pmatrix}$$

$$\mathbf{h}^{(1)} = \tau\left(\mathbf{a}^{(1)}\right) = \begin{pmatrix} 0.0997 \\ 0.0 \\ 0.0997 \end{pmatrix}$$

$$\mathbf{y}'^{(1)} = \mathrm{softmax}\left(\mathbf{Vh}^{(1)} + \mathbf{c}\right) = \mathrm{softmax}\left(\begin{pmatrix} 0.0 & 0.1 & 0.0 \\ -0.2 & 0.0 & 0.0 \end{pmatrix}\begin{pmatrix} 0.0997 \\ 0.0 \\ 0.0997 \end{pmatrix} + \begin{pmatrix} 0.2 \\ 0.1 \end{pmatrix}\right) = \begin{pmatrix} 0.5299 \\ 0.4701 \end{pmatrix}$$

비슷한 방식으로 $t$=2, 3, 4일 때 계산 결과는 다음과 같다.

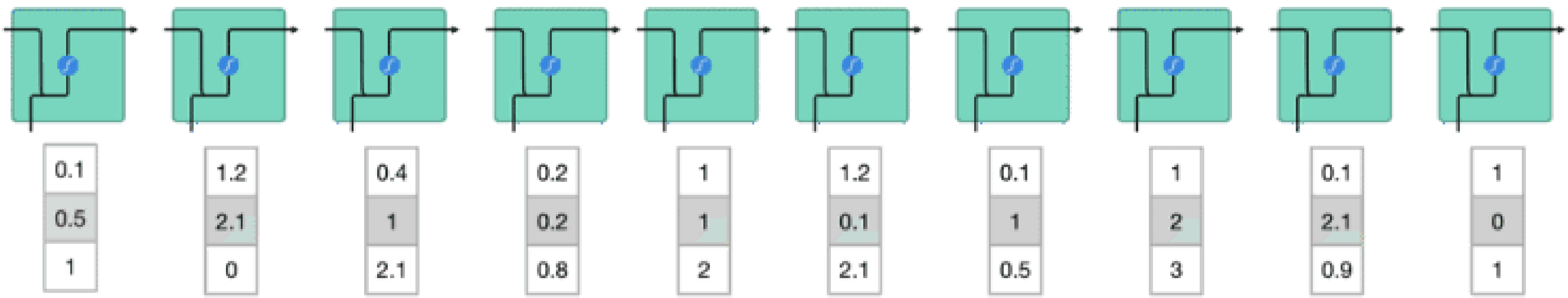
$$\mathbf{y}'^{(2)} = \begin{pmatrix} 0.5260 \\ 0.4740 \end{pmatrix},\ \mathbf{y}'^{(3)} = \begin{pmatrix} 0.5246 \\ 0.4754 \end{pmatrix},\ \mathbf{y}'^{(4)} = \begin{pmatrix} 0.5274 \\ 0.4726 \end{pmatrix}$$

이 샘플의 레이블, 즉 기대 출력이 $\mathbf{y} = \left(\begin{pmatrix} 0.7 \\ 0.4 \end{pmatrix}, \begin{pmatrix} 0.3 \\ 0.4 \end{pmatrix}, \begin{pmatrix} 0.2 \\ 0.4 \end{pmatrix}, \begin{pmatrix} 0.4 \\ 0.5 \end{pmatrix}\right)^{\mathrm{T}}$인데, 출력이 $\mathbf{y}' = \left(\begin{pmatrix} 0.5299 \\ 0.4701 \end{pmatrix}, \begin{pmatrix} 0.5260 \\ 0.4740 \end{pmatrix}, \begin{pmatrix} 0.5246 \\ 0.4754 \end{pmatrix}, \begin{pmatrix} 0.5274 \\ 0.4726 \end{pmatrix}\right)^{\mathrm{T}}$이므로 현재 가중치, 즉 매개변수 $\Theta$는 상당한 오차를 발생시켰다고 판단할 수 있다. 8.2.3절에서는 매개변수 $\Theta$의 값을 반복적으로 개선하여 최적해를 구하는 RNN의 학습 알고리즘을 학습한다.

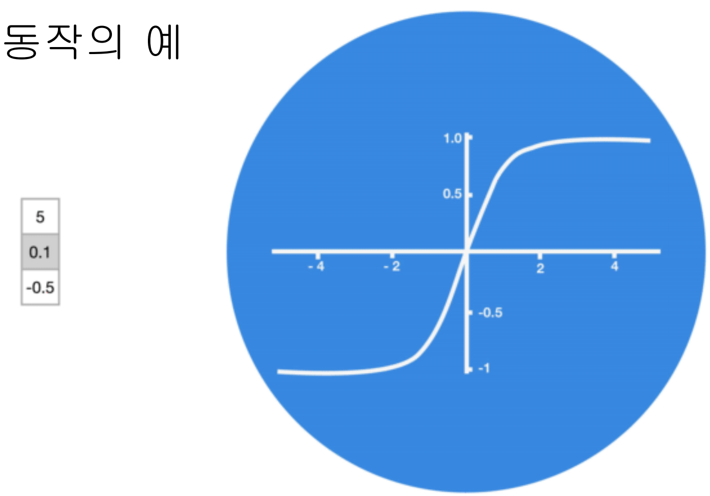
# 8.2.2 동작

## RNN 동작의 예

- 순차적 입력



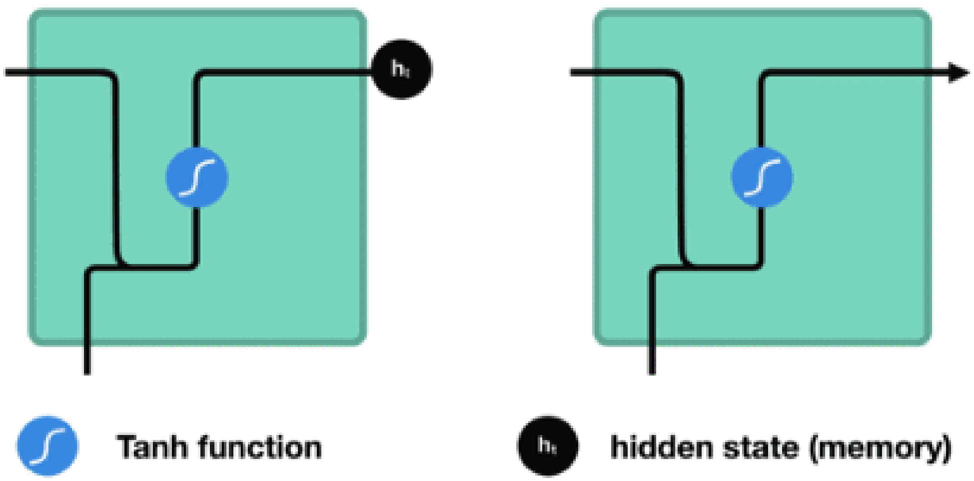

- 비선형 함수 tanh 동작의 예



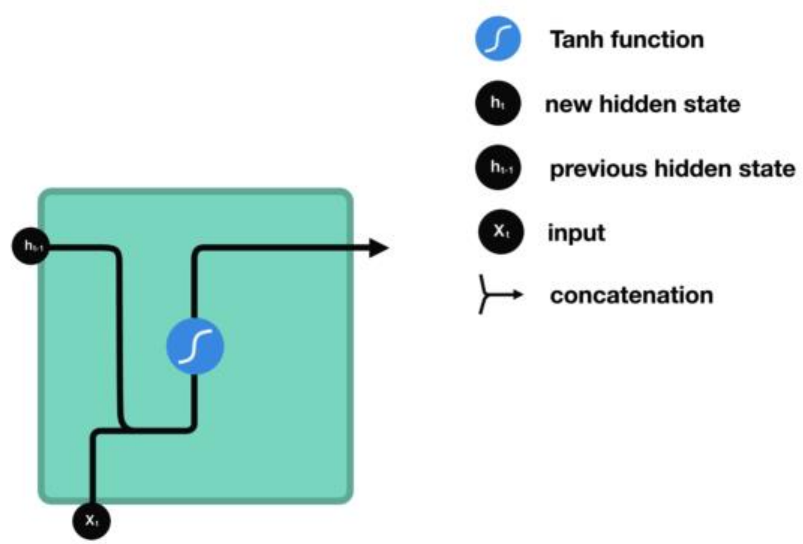
# 8.2.2 동작

## ■ RNN 동작의 예

- 은닉층 정보 전달




- 입력 + 은닉층 정보 갱신



# 8.2.2 동작

## ■ RNN의 기억memory과 문맥 의존성 기능

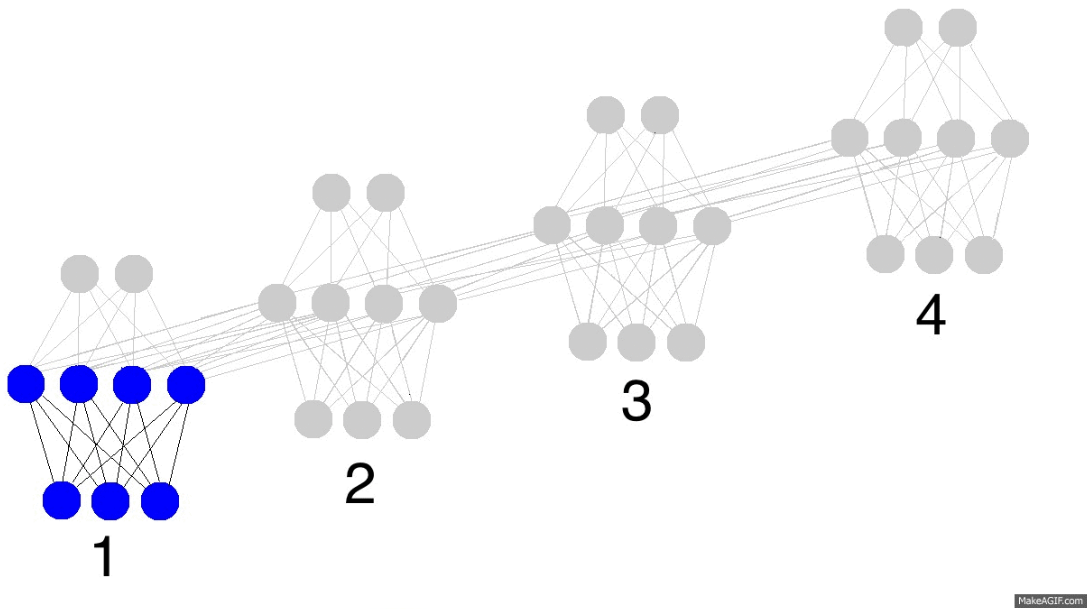
- $\mathbf{x}^{(1)}$이 변하면

  상태 $\mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{h}^{(3)}, \mathbf{h}^{(4)}$ 가 바뀌고, 그에 따라 출력 $\mathbf{y}'^{(1)}, \mathbf{y}'^{(2)}, \mathbf{y}'^{(3)}, \mathbf{y}'^{(4)}$가 바뀜

  → RNN이 $\mathbf{x}^{(1)}$을 기억한다고 말할 수 있음

- 또한 $\mathbf{x}^{(1)}$은 $\mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}^{(4)}$와 상호작용을 한다고 볼 수 있음 → 문맥 의존성

- 기억이 얼마나 지속되는지(장,단기 문맥의존성)는 8.3절에서 다룸



# 8.2.3 BPTT (backpropagation through time) 학습

- 훈련집합 $\mathbb{X}=\{\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_n\}$, $\mathbb{Y}=\{\mathbf{y}_1,\mathbf{y}_2,\cdots,\mathbf{y}_n\}$
  - 샘플 $\mathbf{x}_i$와 $\mathbf{y}_i$는 길이가 $T_i$와 $L_i$인 시간성 데이터

- RNN과 DMLP의 유사성
  - 둘 다 입력층, 은닉층, 출력층을 가짐
  - ([그림 8-8(a)]는 RNN의 노드를 수직으로 배치하여 DMLP와 비교하기 쉽게 함)

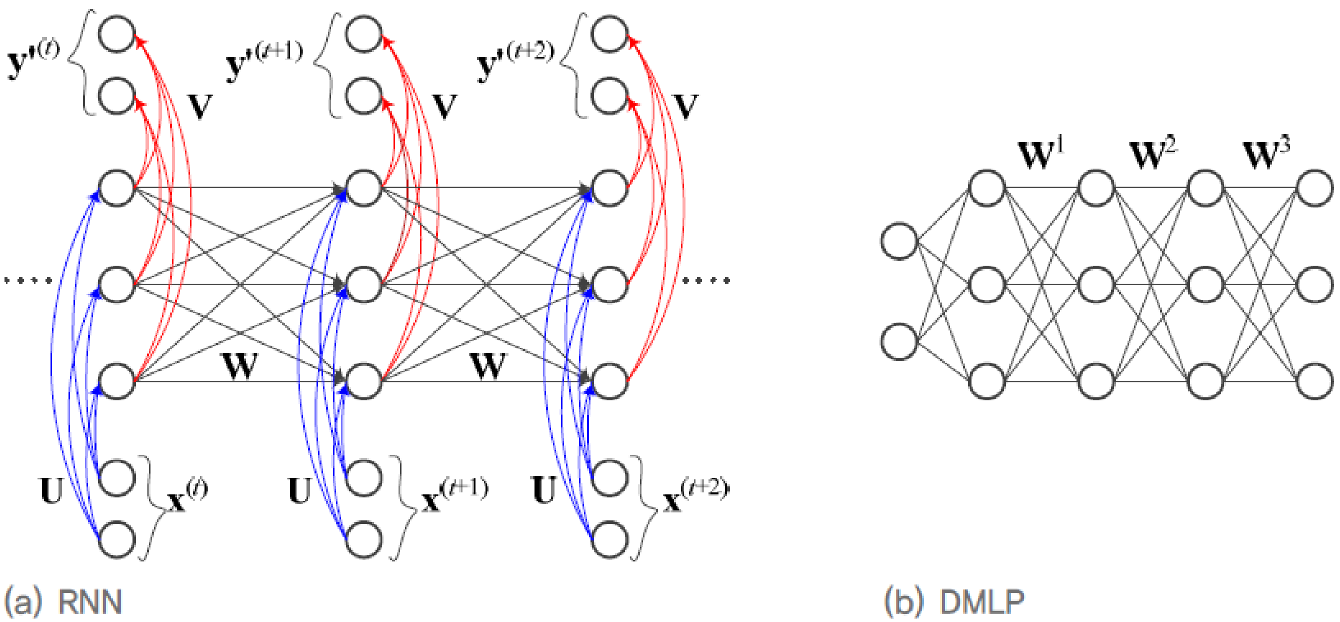



**그림 8-8 RNN과 DMLP의 비교**

# 8.2.3 BPTT 학습

## RNN과 DMLP의 차별성

- RNN은 샘플마다 은닉층의 수가 다름 (얼마나 전달될 수 있는지에 따라 은닉층의 수가 다름)
- DMLP는 왼쪽에 입력, 오른쪽에 출력이 있지만, RNN은 매 순간 입력과 출력이 있음
- RNN은 가중치를 공유함

  → DMLP는 가중치를 $\mathbf{W}^1, \mathbf{W}^2, \mathbf{W}^3, \cdots$로 표기하는데, RNN은 $\mathbf{W}$로 표기

# 8.2.3 BPTT 학습

## 목적함수의 정의

- (예측) 출력 값을 $\mathbf{y}' = \left(y'^{(1)}, y'^{(2)}, \cdots, y'^{(T)}\right)^{\mathrm{T}}$, 목푯값을 $\mathbf{y} = \left(y^{(1)}, y^{(2)}, \cdots, y^{(T)}\right)^{\mathrm{T}}$ 로 표기
  - 평균제곱 오차, 교차 엔트로피, 로그우도 중에 선택하여 사용

$$J(\mathbf{\Theta}) = \sum_{t=1}^{T} J^{(t)}(\mathbf{\Theta}) \tag{8.10}$$

$$\text{평균제곱 오차: } J^{(t)}(\mathbf{\Theta}) = \sum_{j=1}^{q} \left(y_j^{(t)} - y_j'^{(t)}\right)^2 \tag{8.11}$$

$$\text{교차 엔트로피: } J^{(t)}(\mathbf{\Theta}) = -\mathbf{y}^{(t)} \log \mathbf{y}'^{(t)} = -\sum_{j=1}^{q} y_j^{(t)} \log y_j'^{(t)} \tag{8.12}$$

$$\text{로그우도: } J^{(t)}(\mathbf{\Theta}) = -\log y'^{(t)} \tag{8.13}$$

## 학습이 할 일 $\widehat{\mathbf{\Theta}} = \underset{\mathbf{\Theta}}{\operatorname{argmin}} J(\mathbf{\Theta}) = \underset{\mathbf{\Theta}}{\operatorname{argmin}} \sum_{t=1}^{T} J^{(t)}(\mathbf{\Theta})$ (8.14)

# 8.2.3 BPTT 학습

## 경사도 계산

- $\frac{\partial J}{\partial \Theta}$를 구하려면, $\Theta = \{\mathbf{U}, \mathbf{W}, \mathbf{V}, \mathbf{b}, \mathbf{c}\}$이므로 $\frac{\partial J}{\partial \mathbf{U}}, \frac{\partial J}{\partial \mathbf{W}}, \frac{\partial J}{\partial \mathbf{V}}, \frac{\partial J}{\partial \mathbf{b}}, \frac{\partial J}{\partial \mathbf{c}}$ 를 계산해야 함
- 그 중 [그림 8-9]에서처럼 **V**는 출력에만 영향을 미치므로 $\frac{\partial J}{\partial \mathbf{V}}$ 계산이 가장 간단함

→ $\frac{\partial J}{\partial \mathbf{V}}$를 먼저 유도해 봄

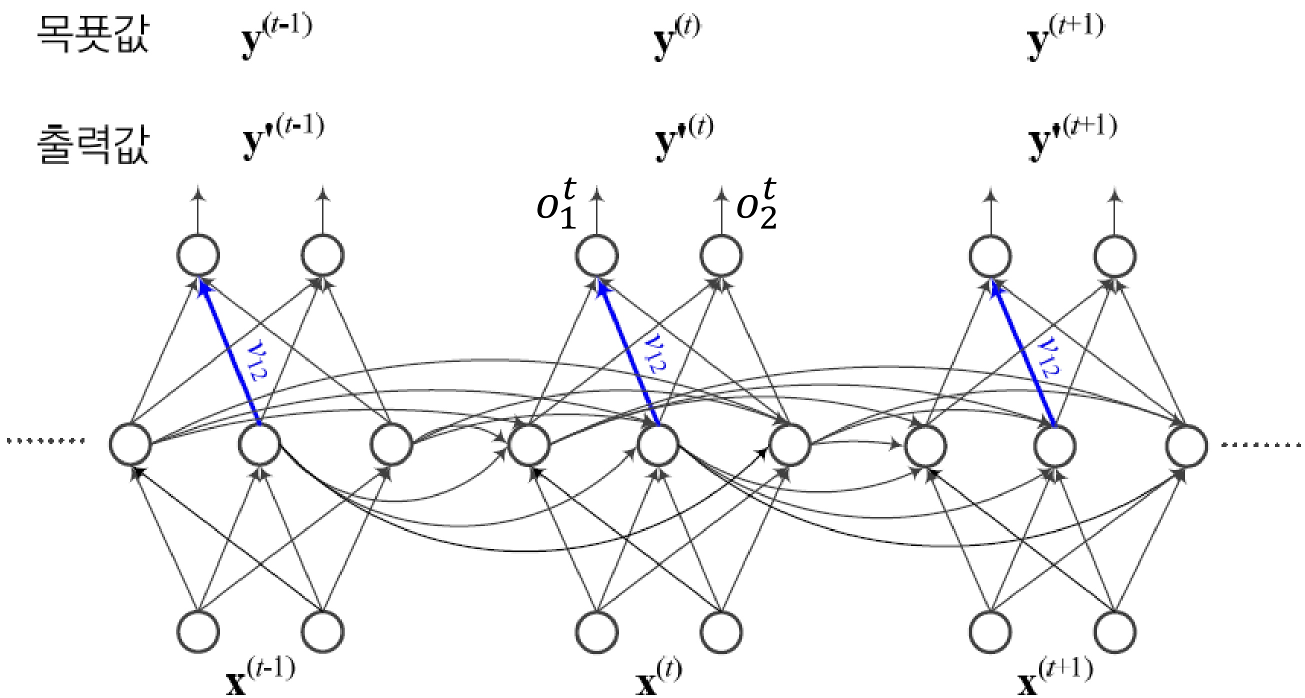
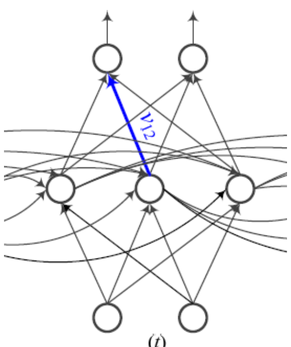



그림 8-9 BPTT 유도를 위한 그레이디언트 계산 예시

# 8.2.3 BPTT 학습

- $\frac{\partial J}{\partial \mathbf{V}}$는 $q*p$ 행렬

$$\frac{\partial J}{\partial \mathbf{V}} = \begin{pmatrix} \frac{\partial J}{\partial v_{11}} & \frac{\partial J}{\partial v_{12}} & \cdots & \frac{\partial J}{\partial v_{1p}} \\ \frac{\partial J}{\partial v_{21}} & \frac{\partial J}{\partial v_{22}} & \cdots & \frac{\partial J}{\partial v_{2p}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial J}{\partial v_{q1}} & \frac{\partial J}{\partial v_{q2}} & \cdots & \frac{\partial J}{\partial v_{qp}} \end{pmatrix} \qquad (8.15)$$




- [그림 8-9]는 $t$ 순간에 $v_{ji}$의 영향을 보여줌 $(j = 1, i = 2)$
- 로그우도를 사용하기로 하고 $v_{12}$로 미분하기 위해 연쇄법칙을 적용하면 ($v_{12}$는 $o_1^t$에만 영향)

$$\frac{\partial J^{(t)}}{\partial v_{12}} = \frac{\partial J^{(t)}}{\partial y'^{(t)}} \frac{\partial y'^{(t)}}{\partial o_1^{(t)}} \frac{\partial o_1^{(t)}}{\partial v_{12}}$$

- 맨 오른쪽 항은 $o_1^{(t)} = \mathbf{v}_1 \mathbf{h}^{(t)} = v_{11} h_1^{(t)} + v_{12} h_2^{(t)} + v_{13} h_3^{(t)}$ 므로 $\frac{\partial o_1^{(t)}}{\partial v_{12}} = h_2^{(t)}$

# 8.2.3 BPTT 학습

- 앞의 2개 항의 계산은

  로그우도를 사용하므로 $\mathbf{y}^{(t)}=(1,0)^{\mathrm{T}}$인 경우와 $\mathbf{y}^{(t)}=(0,1)^{\mathrm{T}}$인 경우로 나누어 생각해야 함

  - $\mathbf{y}^{(t)}=(1,0)^{\mathrm{T}}$인 경우 ($J^{(t)}=-\log y_1'^{(t)}$)를 계산하면,

$$\frac{\partial J^{(t)}}{\partial y'^{(t)}}\frac{\partial y'^{(t)}}{\partial o_1^{(t)}}=\frac{\partial J^{(t)}}{\partial o_1^{(t)}}=\frac{\partial\left(-\log\frac{\exp\left(o_1^{(t)}\right)}{\exp\left(o_1^{(t)}\right)+\exp\left(o_2^{(t)}\right)}\right)}{\partial o_1^{(t)}}$$

$$\leftarrow \quad \mathbf{o}^{(t)}=\mathbf{V}\mathbf{h}^{(t)}+\mathbf{c} \qquad \mathbf{y}'^{(t)}=\mathrm{softmax}(\mathbf{o}^{(t)})$$

$$=\frac{\partial\left(-o_1^{(t)}+\log\left(\exp\left(o_1^{(t)}\right)+\exp\left(o_2^{(t)}\right)\right)\right)}{\partial o_1^{(t)}} \qquad (8.16)$$

$$=-1+\frac{\exp\left(o_1^{(t)}\right)}{\exp\left(o_1^{(t)}\right)+\exp\left(o_2^{(t)}\right)}$$

$$=-1+y_1'^{(t)}$$

  - $\mathbf{y}^{(t)}=(0,1)^{\mathrm{T}}$인 경우 ($J^{(t)}=-\log y_2'^{(t)}$) 도 유도한 다음 두 경우를 같이 쓰면,

$$\left.\begin{aligned}\frac{\partial J^{(t)}}{\partial v_{12}}&=\left(y_1'^{(t)}-1\right)h_2^{(t)},\ \mathbf{y}^{(t)}=(1,0)^{\mathrm{T}}\text{일 때}\\ \frac{\partial J^{(t)}}{\partial v_{12}}&=y_1'^{(t)}h_2^{(t)},\ \mathbf{y}^{(t)}=(0,1)^{\mathrm{T}}\text{일 때}\end{aligned}\right\} \leftarrow \quad y_1'^{(t)}+y_2'^{(t)}=1$$

# 8.2.3 BPTT 학습

- $v_{12}$를 $v_{ji}$로 일반화하고, 2부류를 $q$개 부류로 일반화하면,

$$\left.\begin{array}{l} \dfrac{\partial J^{(t)}}{\partial v_{ji}} = \left(y_j'^{(t)} - 1\right)h_i^{(t)},\ \mathbf{y}^{(t)}\text{의 } j\text{번째 요소가 } 1\text{일 때} \\ \dfrac{\partial J^{(t)}}{\partial v_{ji}} = y_j'^{(t)} h_i^{(t)},\ \mathbf{y}^{(t)}\text{의 } j\text{번째 요소가 } 0\text{일 때} \end{array}\right\}$$

- 좀 더 간결하게 표현하면,

$$\frac{\partial J^{(t)}}{\partial v_{ji}} = \left(y_j'^{(t)} - y_j^{(t)}\right)h_i^{(t)} \tag{8.17}$$

- $1, 2, \cdots, T$ 순간을 모두 고려하면,

$$\frac{\partial J}{\partial v_{ji}} = \sum_{t=1}^{T}\left(y_j'^{(t)} - y_j^{(t)}\right)h_i^{(t)} \tag{8.18}$$

# 8.2.3 BPTT 학습

## BPTT (back-propagation through time) 알고리즘

- $v_{ji}$로 미분하는 식 (8.18)을 행렬 전체를 위한 식 $\frac{\partial J}{\partial \mathbf{V}}$로 확장하고,
  $\frac{\partial J}{\partial \mathbf{U}}, \frac{\partial J}{\partial \mathbf{W}}, \frac{\partial J}{\partial \mathbf{b}}, \frac{\partial J}{\partial \mathbf{c}}$까지 유도하면 BPTT가 완성됨
- 이 확장 작업에 필요한 식 (8.16)을 벡터 형태로 일반화하면,

$$\frac{\partial J^{(t)}}{\partial \mathbf{o}^{(t)}} = \mathbf{y}'^{(t)} - \mathbf{y}^{(t)} \qquad (8.19)$$

## 은닉층에서의 미분

- 순간 $t$의 은닉층값 $\mathbf{h}^{(t)}$의 미분은
  그 이후의 은닉층과 출력층에 영향을 주므로 $\mathbf{V}$로 미분하는 것보다 복잡
- 우선 이후가 없는 마지막 순간 $T$에 대해 미분식을 유도하면,

$$\frac{\partial J^{(T)}}{\partial \mathbf{h}^{(T)}} = \frac{\partial J^{(T)}}{\partial \mathbf{o}^{(T)}} \frac{\partial \mathbf{o}^{(T)}}{\partial \mathbf{h}^{(T)}} = \mathbf{V}^{\mathrm{T}} \frac{\partial J^{(T)}}{\partial \mathbf{o}^{(T)}} \qquad (8.20)$$

$\longleftarrow \mathbf{o}^{(t)} = \mathbf{V}\mathbf{h}^{(t)} + \mathbf{c}$

# 8.2.3 BPTT 학습

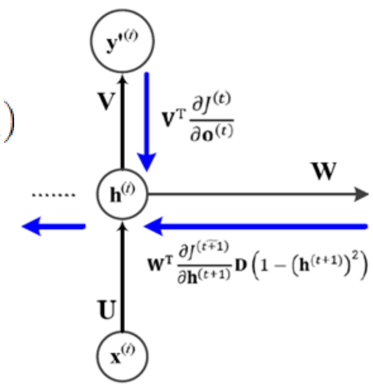
- $T$-1 순간의 경사도를 유도하면,
  - $\mathbf{D}\left(1-\left(\mathbf{h}^{(T)}\right)^2\right)$는 $i$번 열의 대각선이 $1-\left(h_i^{(T)}\right)^2$을 가진 대각 행렬

$$\frac{\partial\left(J^{(T-1)}+J^{(T)}\right)}{\partial \mathbf{h}^{(T-1)}}=\frac{\partial J^{(T-1)}}{\partial \mathbf{o}^{(T-1)}}\frac{\partial \mathbf{o}^{(T-1)}}{\partial \mathbf{h}^{(T-1)}}+\frac{\partial \mathbf{h}^{(T)}}{\partial \mathbf{h}^{(T-1)}}\frac{\partial J^{(T)}}{\partial \mathbf{h}^{(T)}}$$

$$=\mathbf{V}^{\mathrm{T}}\frac{\partial J^{(T-1)}}{\partial \mathbf{o}^{(T-1)}}+\mathbf{W}^{\mathrm{T}}\frac{\partial J^{(T)}}{\partial \mathbf{h}^{(T)}}\mathbf{D}\left(1-\left(\mathbf{h}^{(T)}\right)^2\right)$$

- $t$ 순간으로 일반화하면, 경사도를 역전파하는 순환식인 식 (8.21)을 얻음
  - $J^{(\tilde{t})}$는 $t$를 포함하여 이후 목적함수의 값을 모두 더한 값, 즉 $J^{(\tilde{t})}=J^{(t)}+J^{(t+1)}+\cdots+J^{(T)}$

$$\frac{\partial J^{(\tilde{t})}}{\partial \mathbf{h}^{(t)}}=\mathbf{V}^{\mathrm{T}}\frac{\partial J^{(t)}}{\partial \mathbf{o}^{(t)}}+\mathbf{W}^{\mathrm{T}}\frac{\partial J^{(\widetilde{t+1})}}{\partial \mathbf{h}^{(t+1)}}\mathbf{D}\left(1-\left(\mathbf{h}^{(t+1)}\right)^2\right) \qquad (8.21)$$



# 8.2.3 BPTT 학습

- [그림 8-10]은 식 (8.21)을 설명

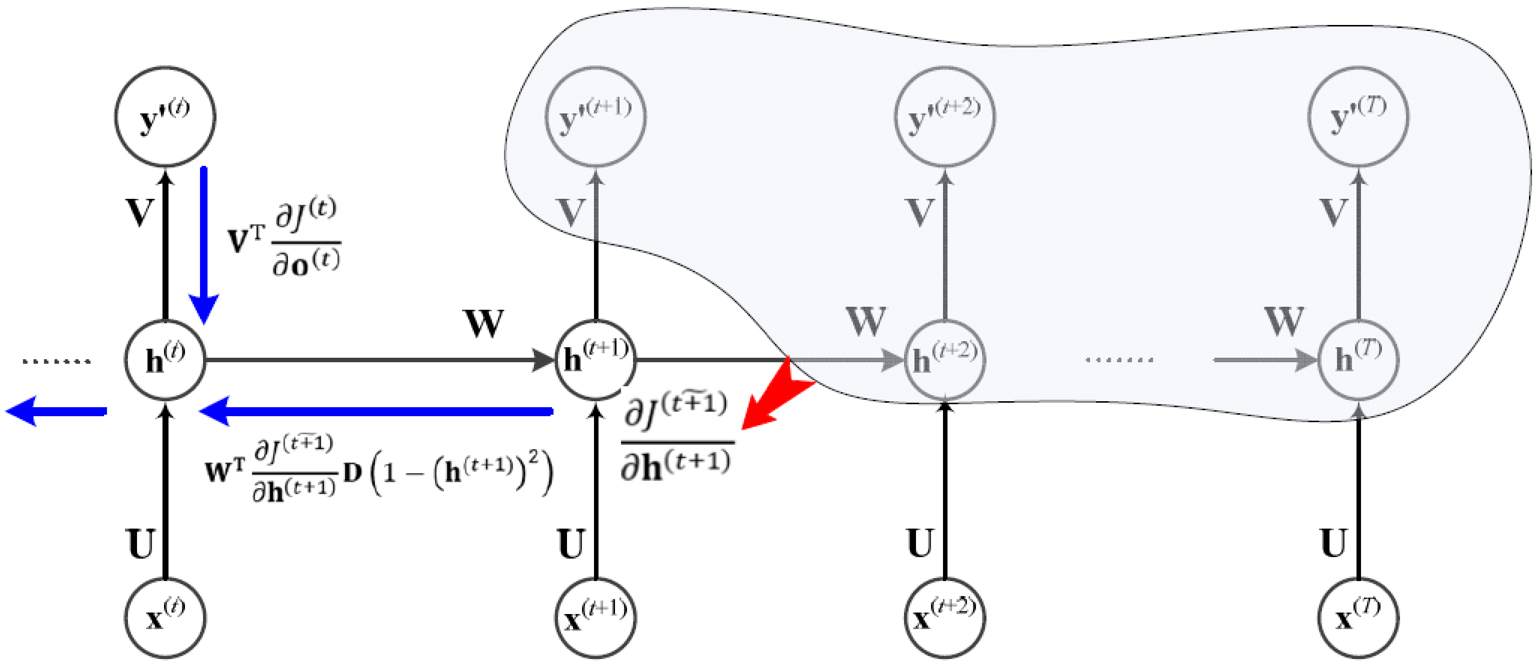



그림 8-10 역전파 순환식으로서 식 (8.21)의 동작

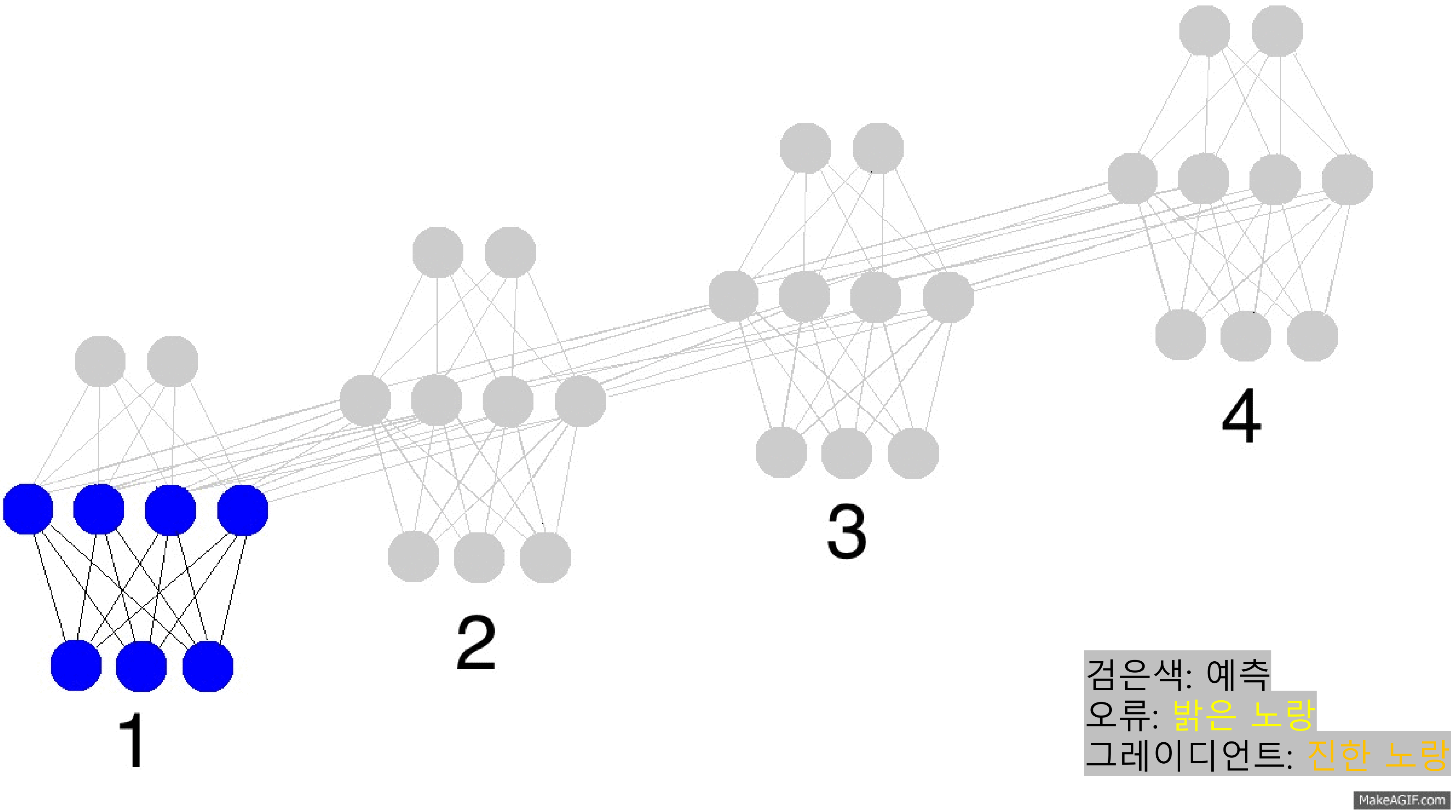
# 8.2.3 BPTT 학습

- BPTT 알고리즘

$$
\begin{cases}
\dfrac{\partial J}{\partial \mathbf{V}} = \displaystyle\sum_{t=1}^{T} \dfrac{\partial J^{(t)}}{\partial \mathbf{o}^{(t)}} \mathbf{h}^{(t)\mathrm{T}} & (8.22) \\[2ex]
\dfrac{\partial J}{\partial \mathbf{W}} = \displaystyle\sum_{t=1}^{T} \mathbf{D}\left(1 - \left(\mathbf{h}^{(t)}\right)^2\right) \dfrac{\partial J^{(\tilde{t})}}{\partial \mathbf{h}^{(t)}} \mathbf{h}^{(t-1)\mathrm{T}} & (8.23) \\[2ex]
\dfrac{\partial J}{\partial \mathbf{U}} = \displaystyle\sum_{t=1}^{T} \mathbf{D}\left(1 - \left(\mathbf{h}^{(t)}\right)^2\right) \dfrac{\partial J^{(\tilde{t})}}{\partial \mathbf{h}^{(t)}} \mathbf{x}^{(t)\mathrm{T}} & (8.24) \\[2ex]
\dfrac{\partial J}{\partial \mathbf{c}} = \displaystyle\sum_{t=1}^{T} \dfrac{\partial J^{(t)}}{\partial \mathbf{o}^{(t)}} & (8.25) \\[2ex]
\dfrac{\partial J}{\partial \mathbf{b}} = \displaystyle\sum_{t=1}^{T} \mathbf{D}\left(1 - \left(\mathbf{h}^{(t)}\right)^2\right) \dfrac{\partial J^{(\tilde{t})}}{\partial \mathbf{h}^{(t)}} & (8.26)
\end{cases}
$$

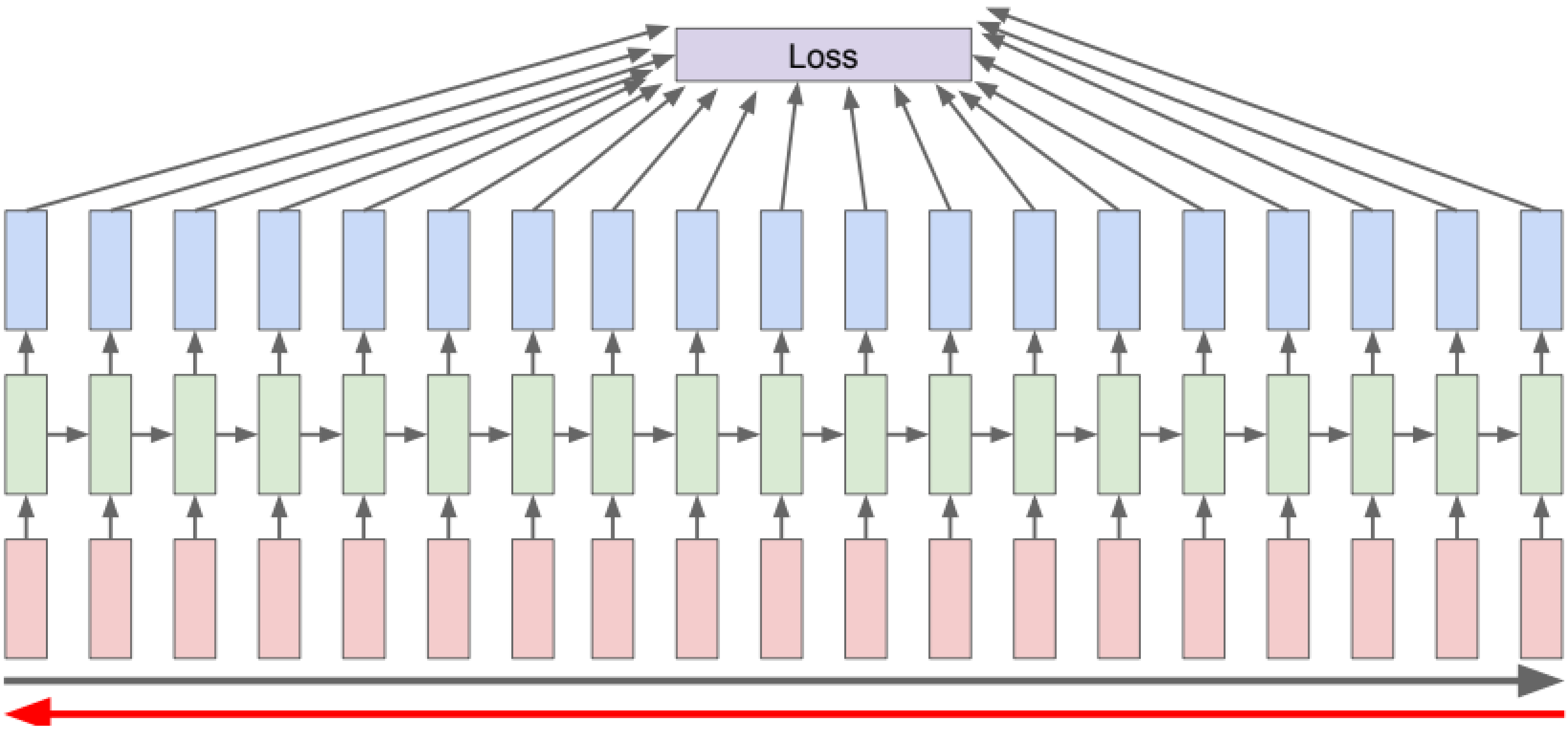
# 8.2.3 BPTT 학습

- BPTT 학습: 전방 계산과 오류 역전파 수행



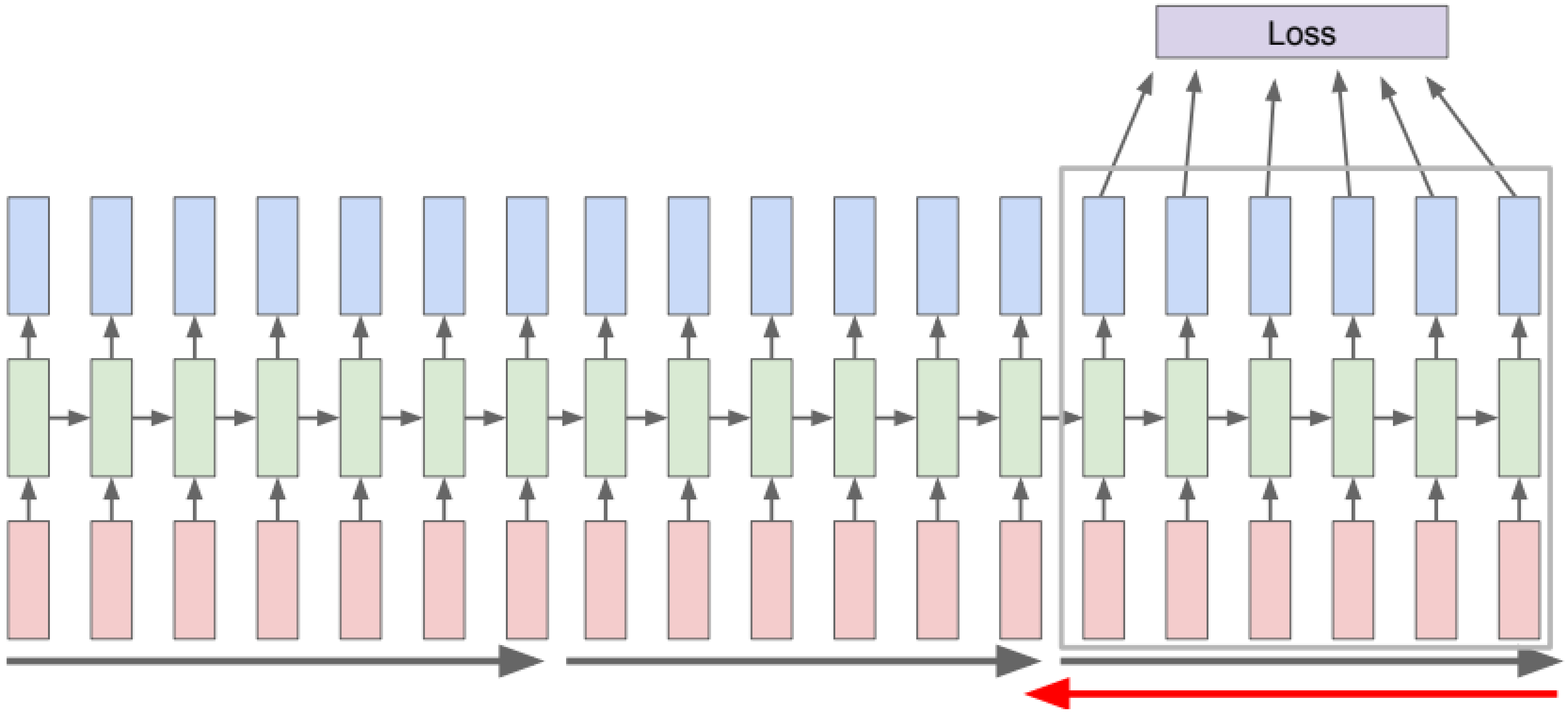
# 8.2.3 BPTT 학습

- 시간에 따른 오류역전파의 동작



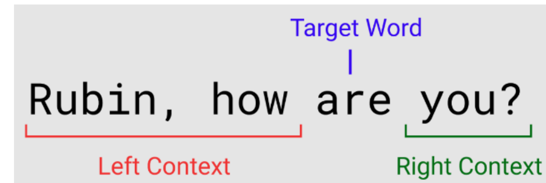
# 8.2.3 BPTT 학습

- 잘린truncated 시간에 따른 오류역전파의 동작



# 8.2.4 양방향 RNN

- 양방향 문맥 의존성
  - 왼쪽에서 오른쪽으로만 정보가 흐르는 단방향 RNN은 한계
  - 예, [그림 8-11]에서
    ‘거지’와 ‘지지’를 구별하기 어려움




거지 같은 행색

꺼지지 않는

그림 8-11 양방향 문맥 의존성

- 양방향 RNN (Bidirectional RNN)
  - $t$ 순간의 단어는
    앞쪽 단어와 뒤쪽 단어 정보를 모두 보고 처리됨
  - 기계 번역에서도 BRNN을 활용함
    ← 8.5.2절

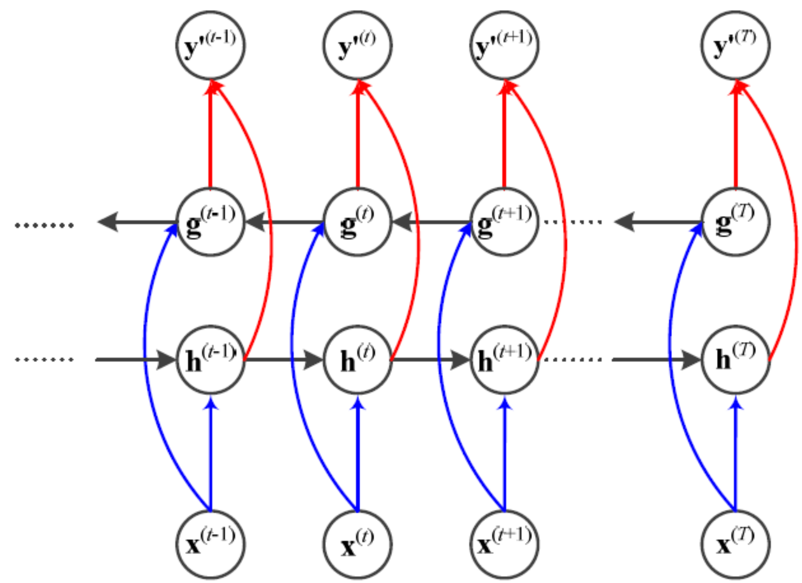



그림 8-12 양방향 RNN(BRNN)의 구조와 동작

# 8.2.4 양방향 RNN

- 양방향 RNN (Bidirectional RNN) 예