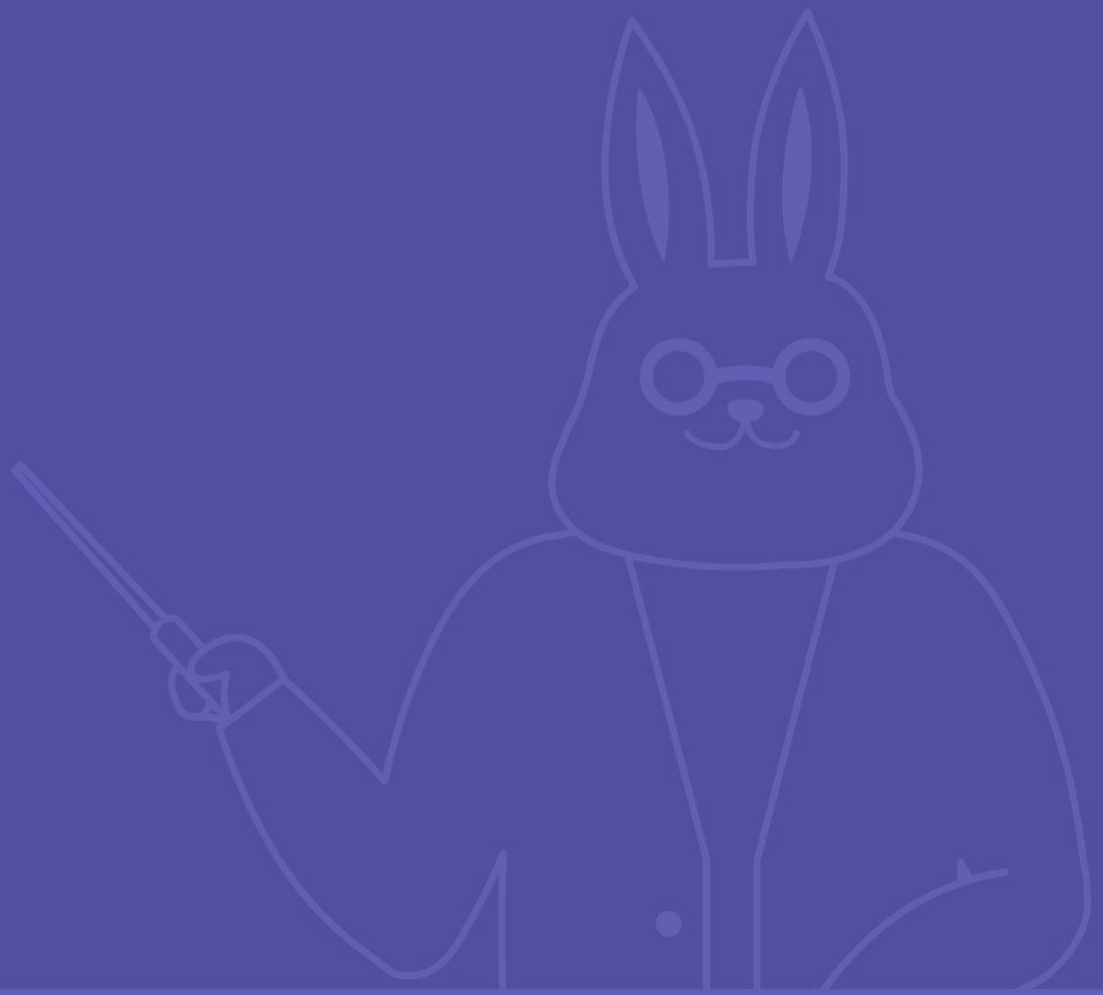
CNN/RNN

03 Recurrent Neural Network

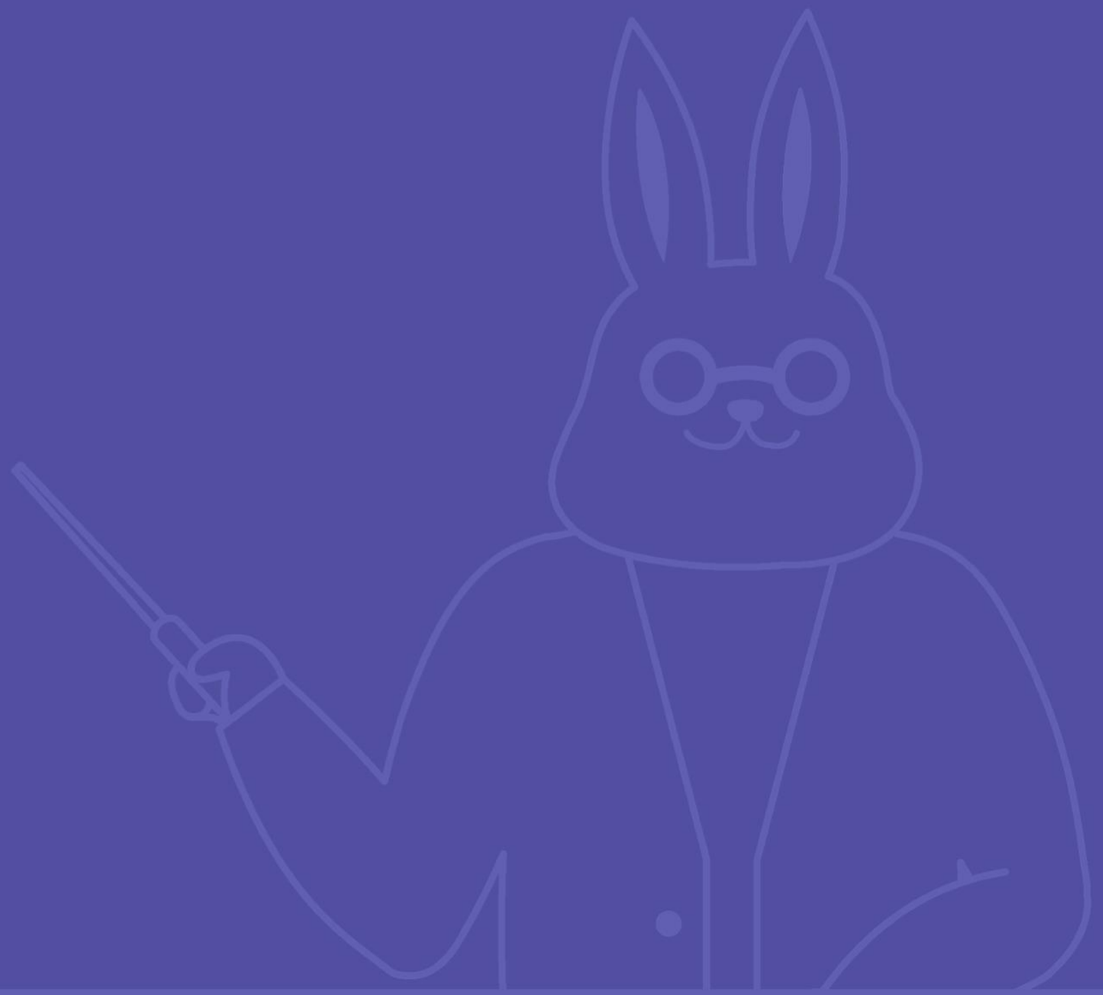


/* elice */

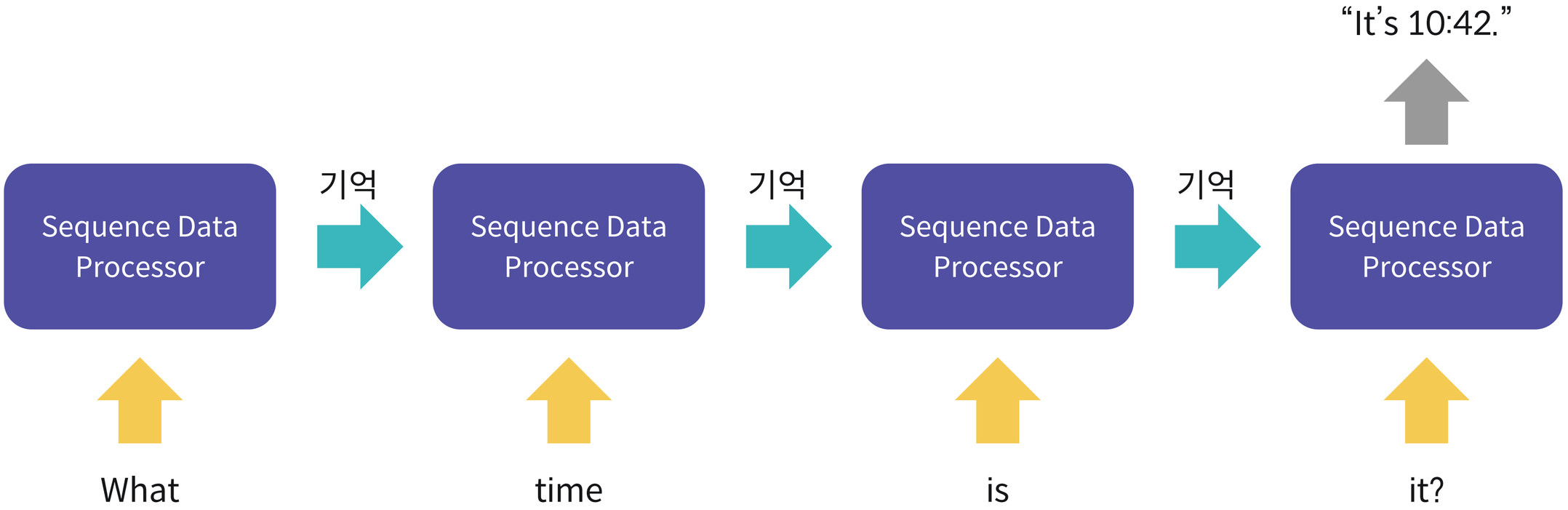
목차

01

순차 데이터란?

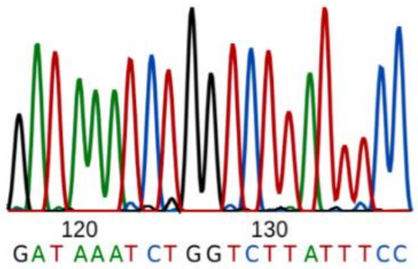


왜 RNN과 순차 데이터인가?



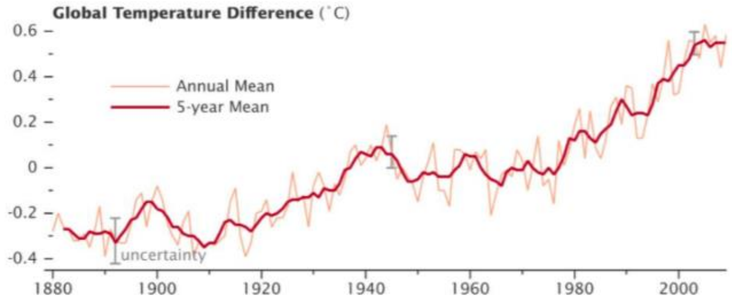
- RNN(Recurrent Neural Network)은 CNN과 함께 대표적인 딥러닝 모델
- 시계열 데이터 같은 **순차 데이터(Sequential Data)** 처리를 위한 모델
- RNN의 이해에는 **순차 데이터가 가지는 특징**의 이해가 필요

순차 데이터란?



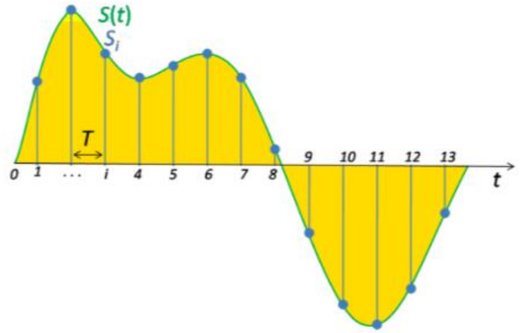
DNA 염기 서열

(Sequential Data)



세계 기온 변화

(Temporal Sequence)

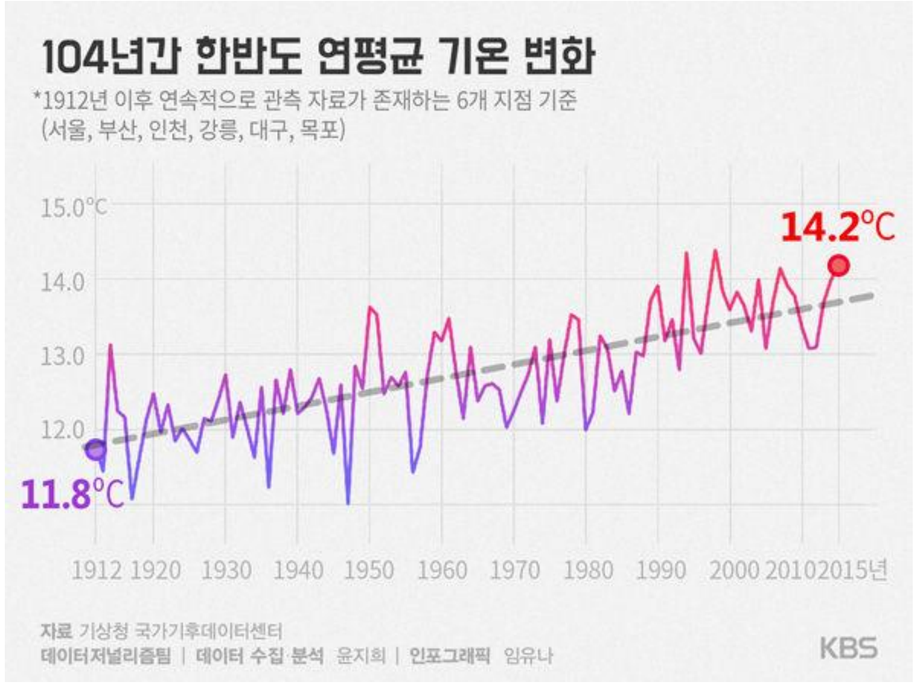


샘플링된 소리 신호

(Time Series)

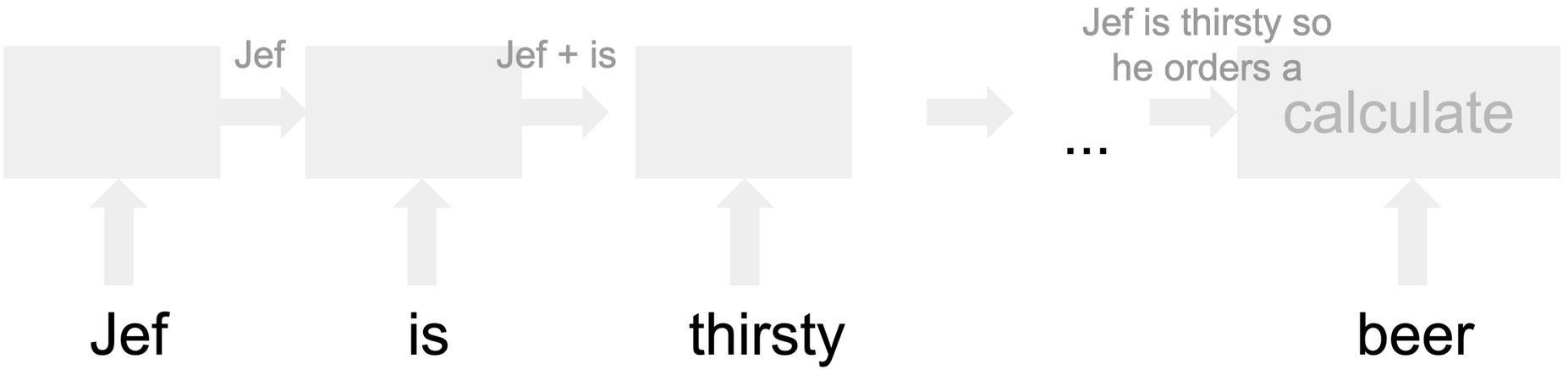
- **순서(Order)**를 가지고 나타나는 데이터
- **날짜에 따른 기온 데이터**나 **단어들로 이루어진 문장** 등으로 유명
  - 데이터 내 **각 개체 간의 순서**가 중요

시계열 데이터 (Time-Series Data)



- **일정한 시간 간격**을 가지고 얻어낸 데이터
- 예) 연도별 대한민국의 평균 기온, 시간별 주식 가격 기록 등

자연어 데이터 (Natural Language)



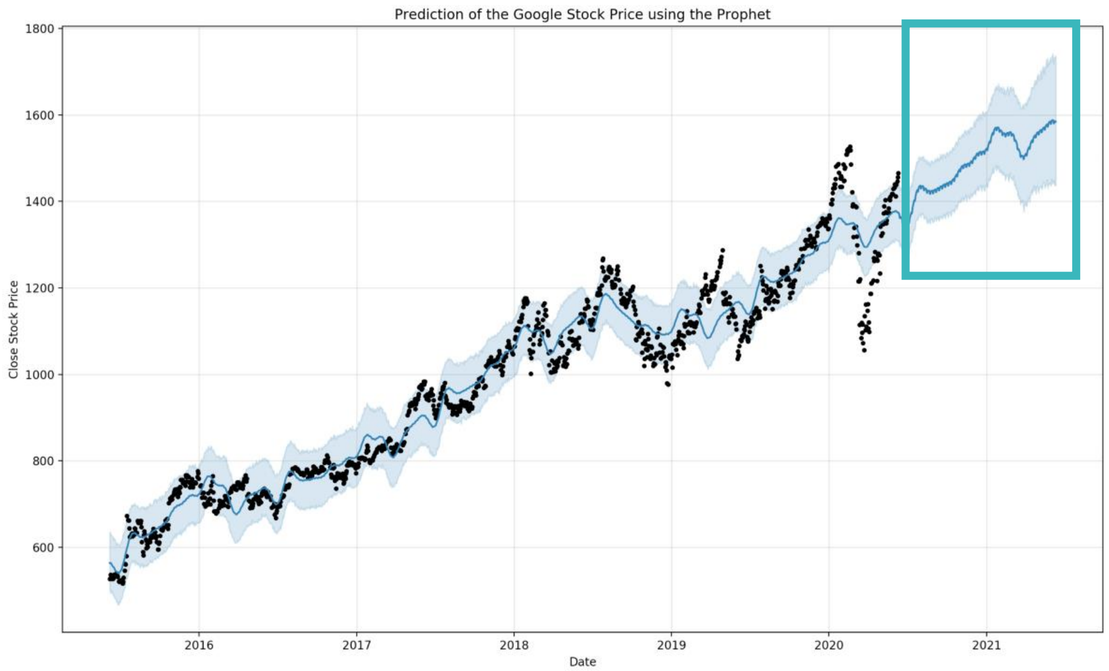
- 인류가 말하는 **언어**를 의미
- 주로 문장 내에서 **단어가 등장하는 순서**에 주목

02

딥러닝을 활용한 순차 데이터 처리 예시

경향성 파악

예측된 주가



예측된 기온

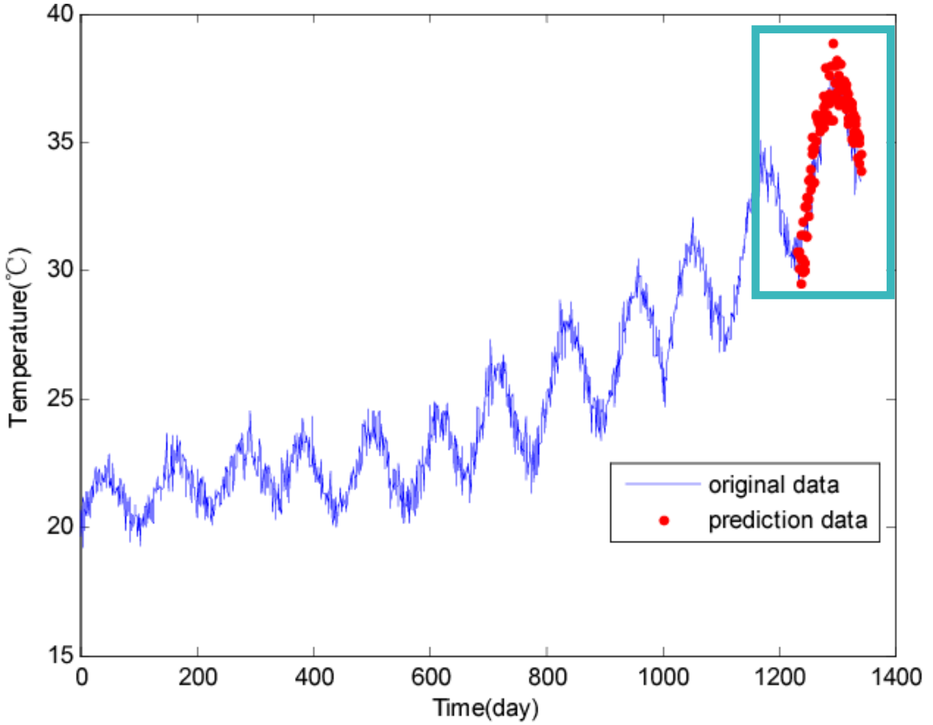
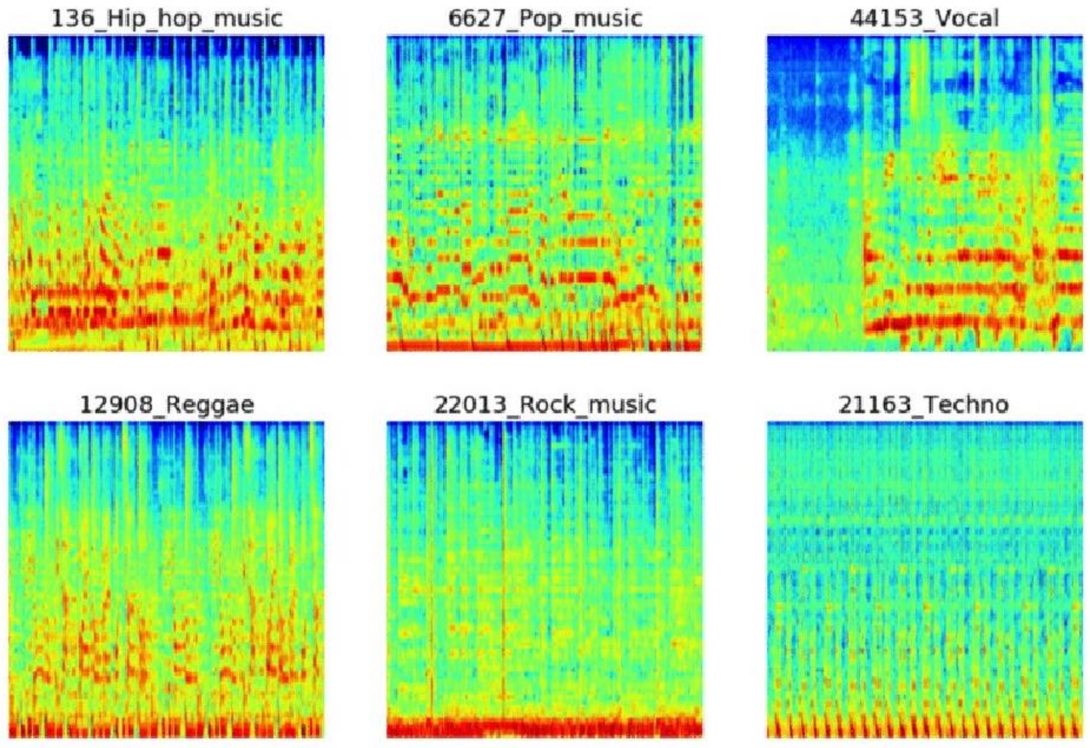
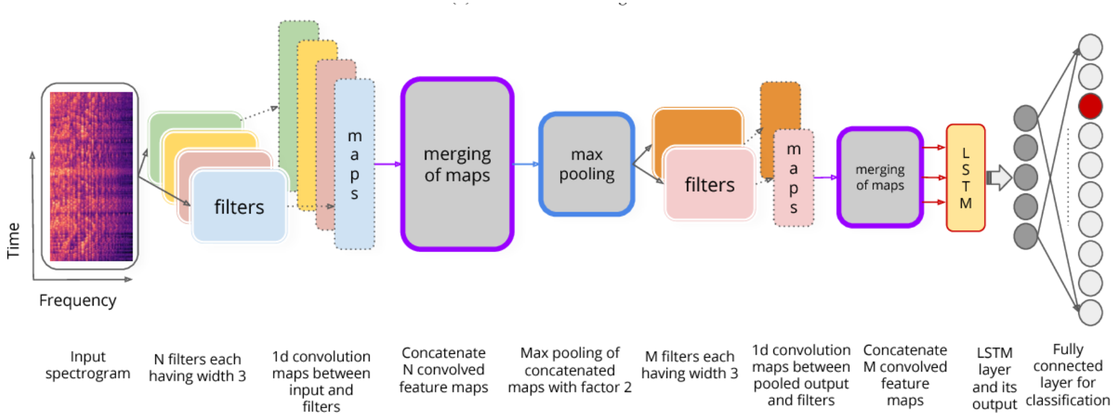


Figure 5. The forecasting result of the method based on EMD and SVM.

- 주가 예측
- 기온 예측
- 이외에도 다양한 시계열 특징을 가지는 데이터에 적용 가능

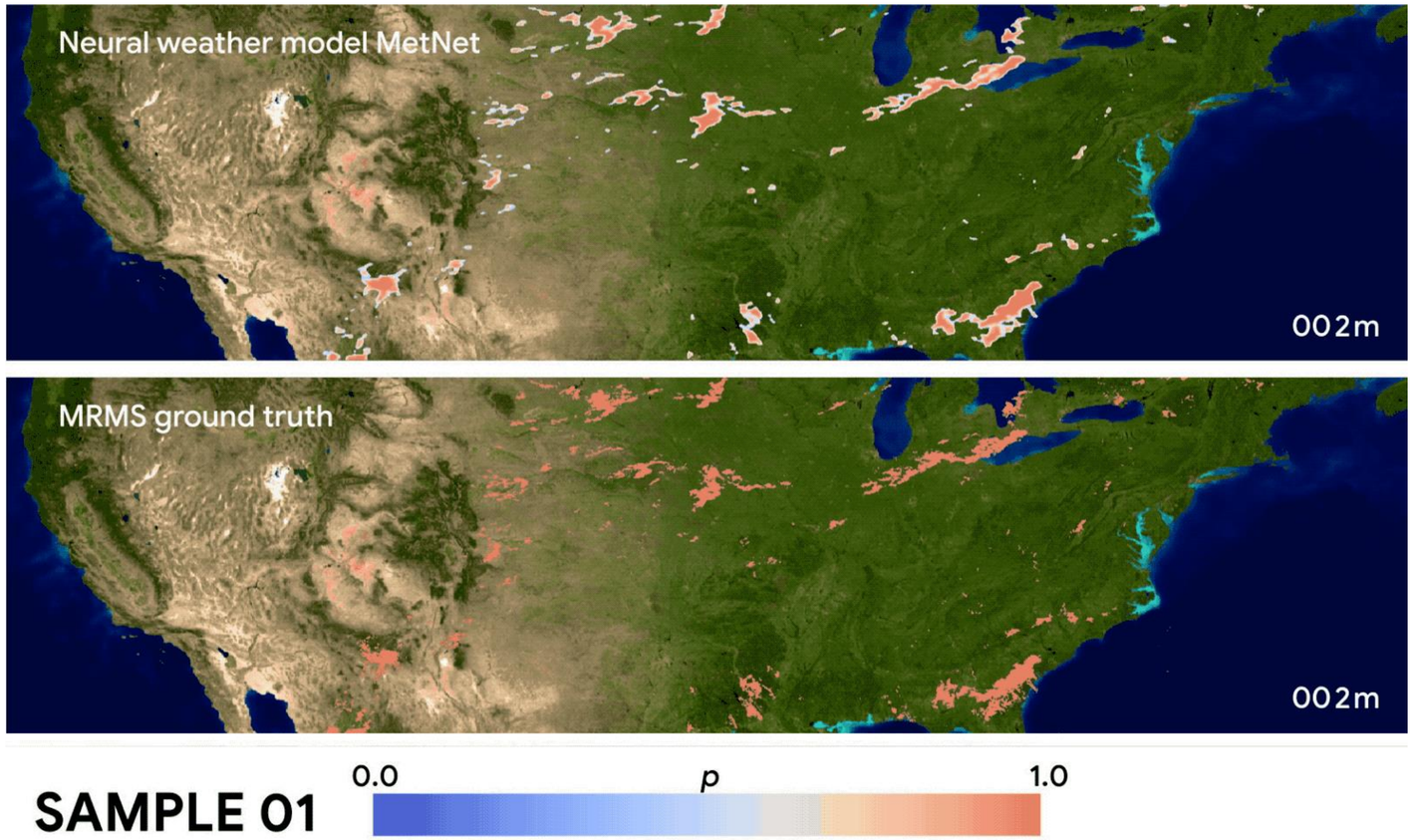
음악 장르 분석





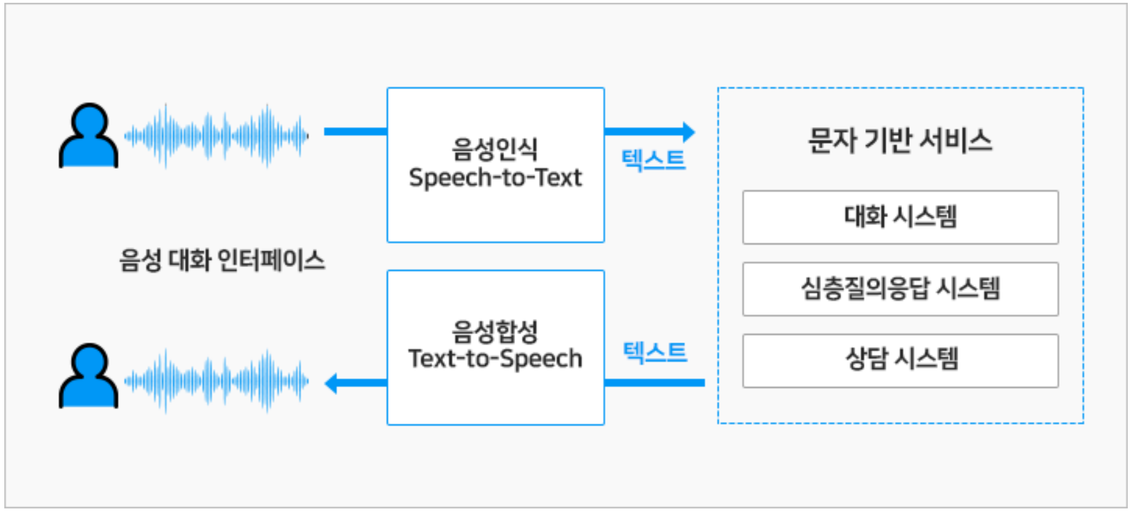
- 오디오 파일은 본질적으로 시계열 데이터
- 음파 형태 등을 분석하여 **오디오 파일의 장르**를 분석

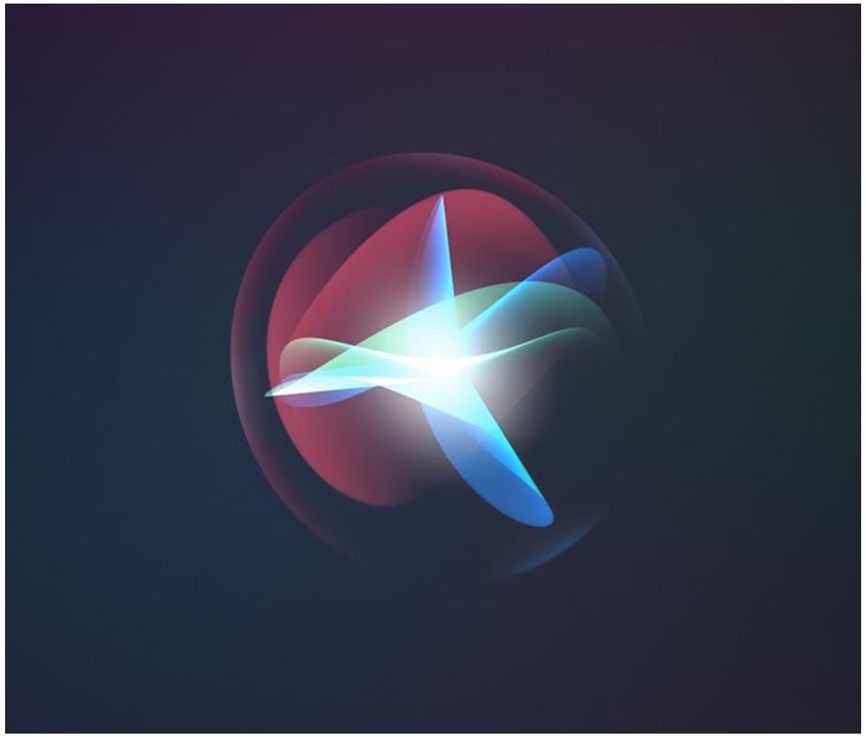
강수량 예측 (Precipitation Forecasting)



구글에서 이미지 처리 기술과 결합하여 주도적으로 연구 (예: MetNet)

음성 인식 (Speech Recognition)



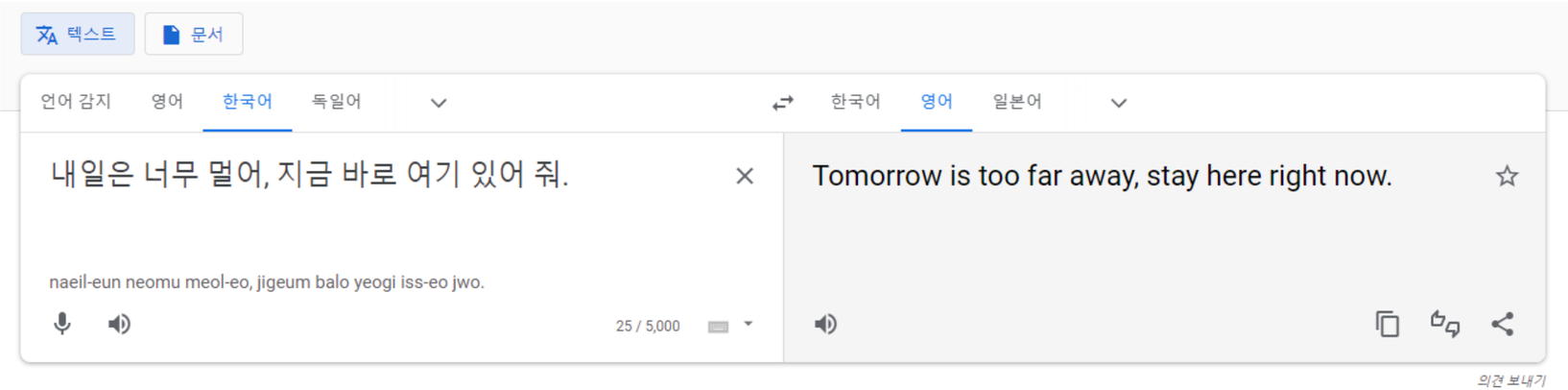


Apple Siri

음성에 포함된 단어나 소리를 추출

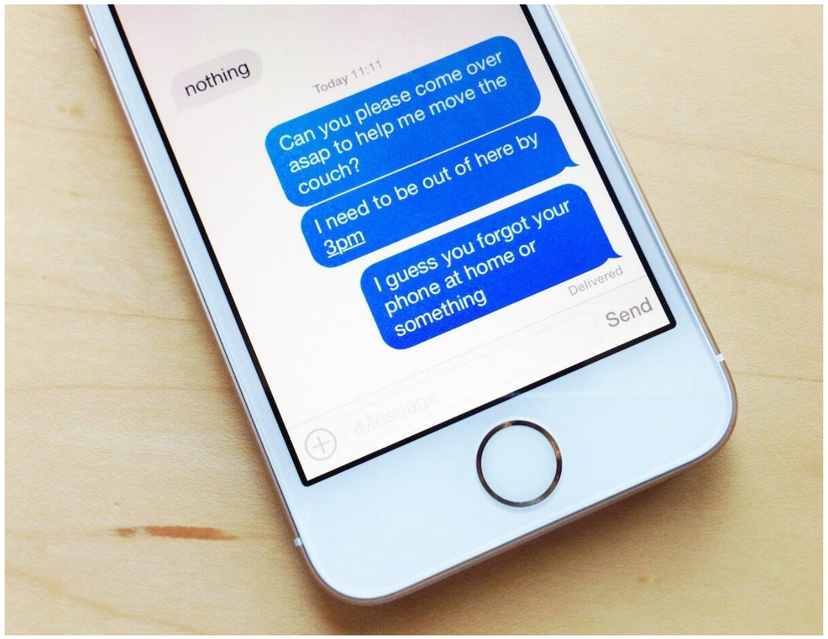
예) Apple Siri, Google Assistant

번역기 (Translator)

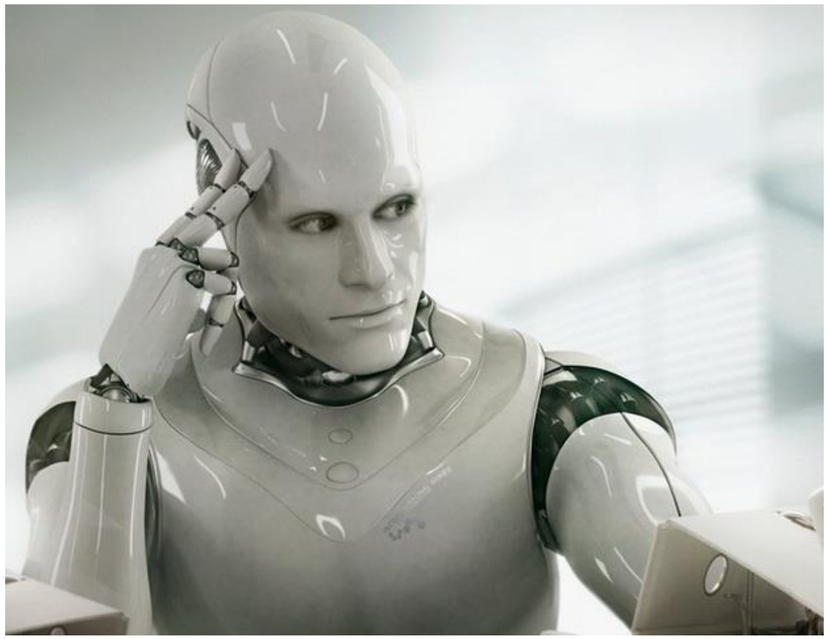


- 두 언어 간 문장 번역을 수행
- 딥러닝의 발전 이후 번역의 자연스러움이 향상
- 이미지 처리와 결합하여 **실시간 번역**도 제공
- 예) 구글 번역, 네이버 파파고 등

챗봇 (Chatbot)



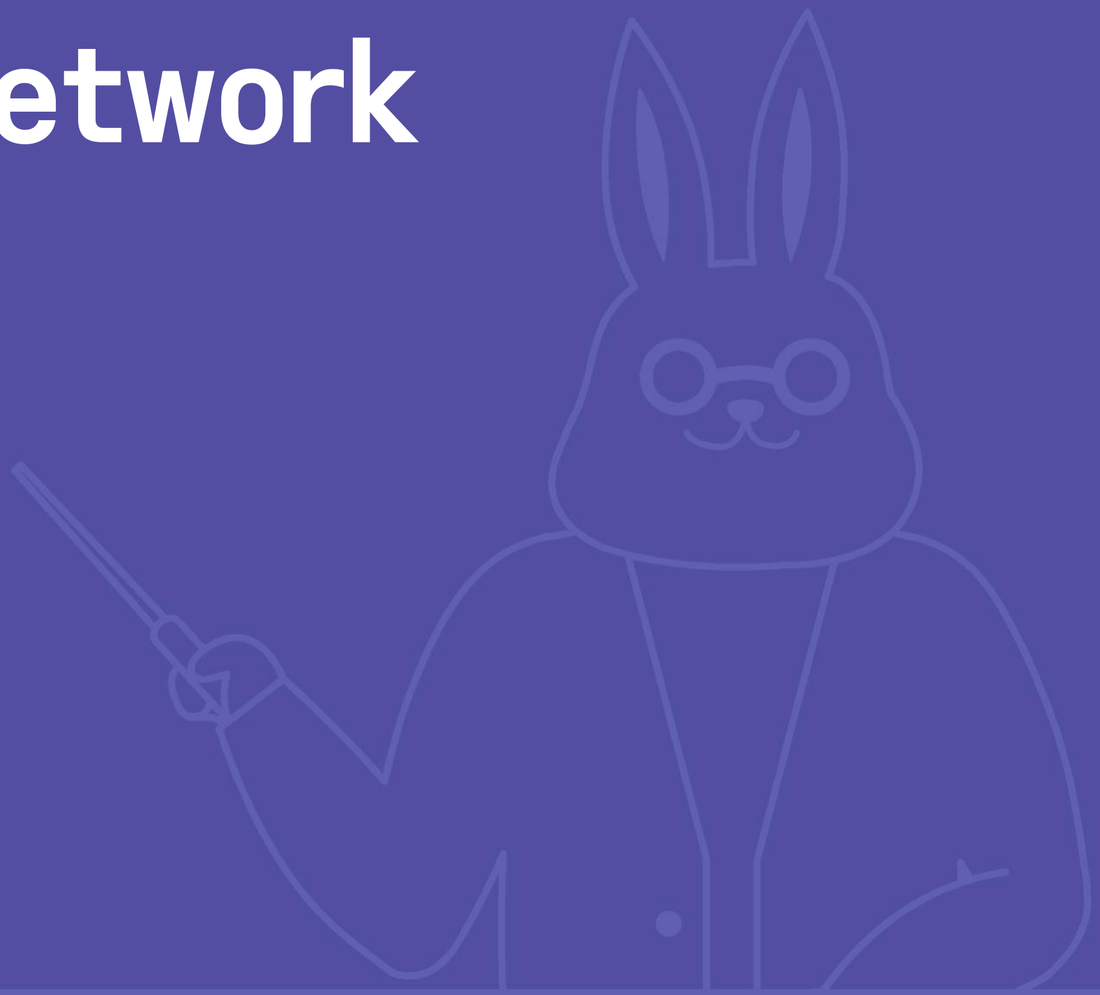
+



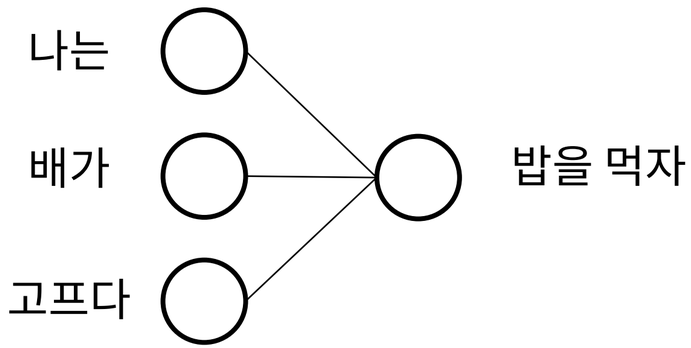
- 사용자의 질문에 사람처럼 응답하고자 하는 프로그램
- 사용자의 **질문을 분석** 후 질문에 적절한 **응답을 생성**

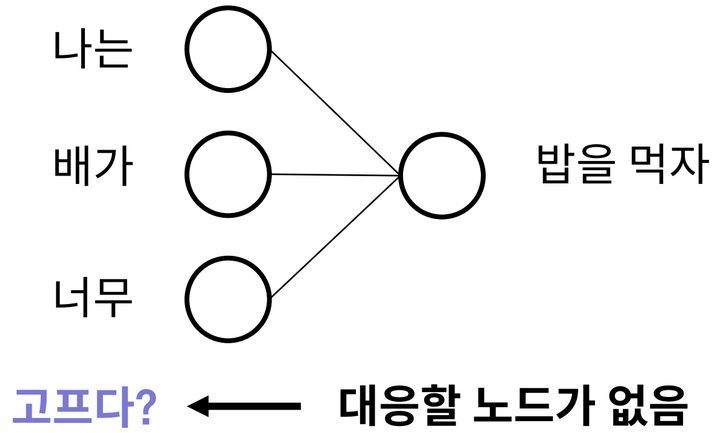
03

Recurrent Neural Network



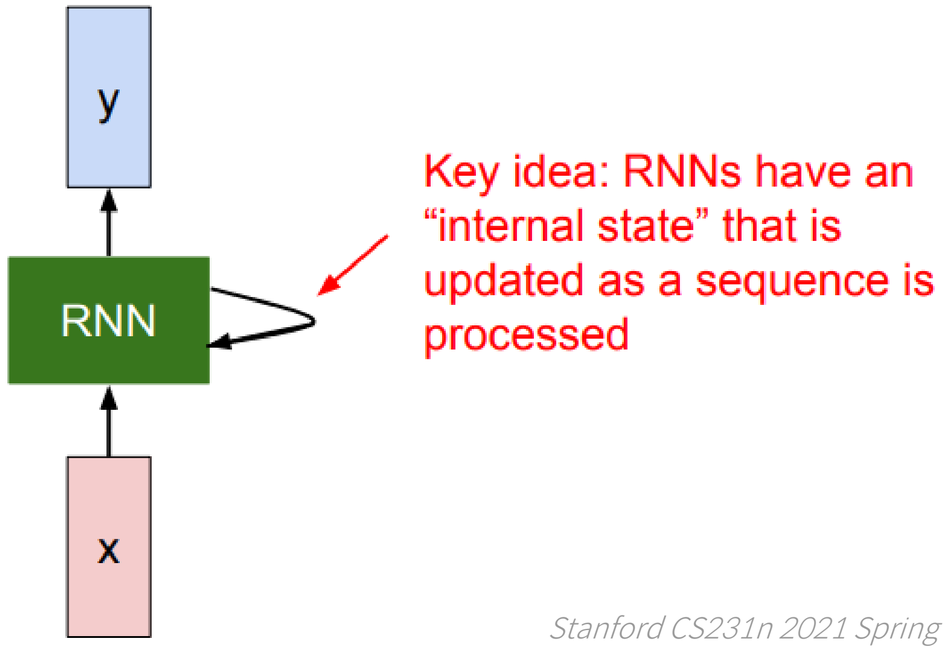
Fully-connected Layer와 순차 데이터





- FC Layer는 입력 노드 개수와 출력 노드 개수가 **정해짐**
- 순차 데이터는 하나의 데이터를 이루는 **개체 수가 다를 수 있음**
  - 문장은 모두 서로 다른 개수의 단어로 이루어짐
- 또한 FC Layer는 **순서 고려가 불가능**

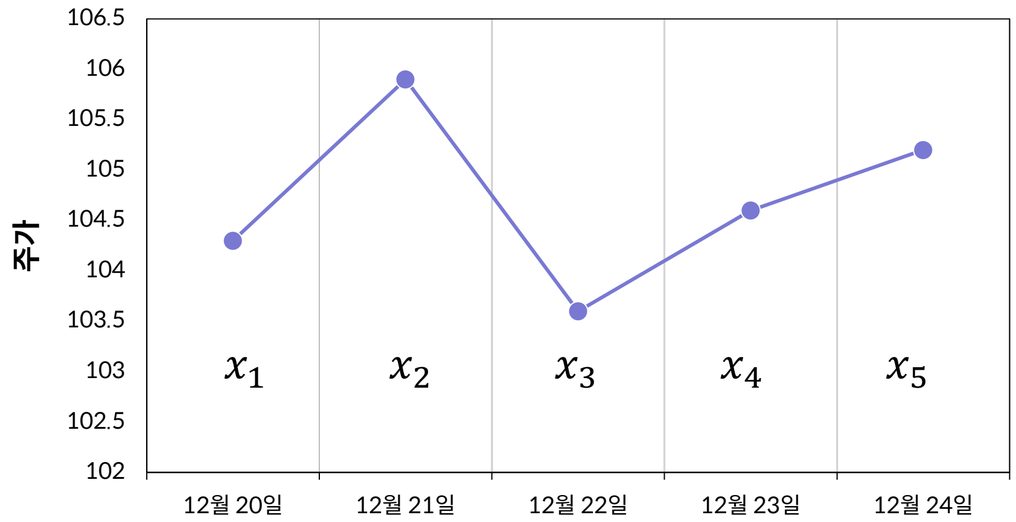
Recurrent Neural Network

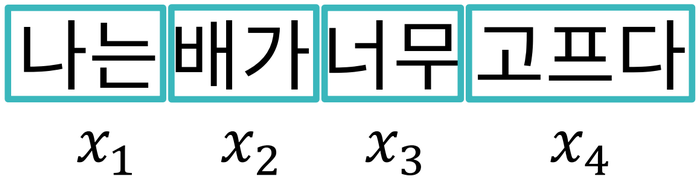


Stanford CS231n 2021 Spring

- 따라서 순차 데이터 처리를 위한 딥러닝 모델이 등장
- RNN의 대표적인 구성 요소
  - **Hidden State**: 순환 구조를 구현하는 핵심 장치

입력 데이터 구조





- $x_1, x_2, x_3, \ldots, x_n$과 같이 데이터의 나열
- 각 x_t의 의미
  - **시계열 데이터**: 일정 시간 간격으로 나눠진 데이터 개체 하나
  - **자연어 데이터**: 문장 내의 각 단어

시계열 데이터의 벡터 변환

	p (mbar)	T (degC)	Tpot (K)	Tdew (degC)	rh (%)	VPmax (mbar)	VPact (mbar)	VPdef (mbar)	sh (g/kg)	H2OC (mmol/mol)	rho (g/m**3)	wv (m/s)	max. wv (m/s)	wd (deg)
5	996.50	-8.05	265.38	-8.78	94.4	3.33	3.14	0.19	1.96	3.15	1307.86	0.21	0.63	192.7
11	996.62	-8.88	264.54	-9.77	93.2	3.12	2.90	0.21	1.81	2.91	1312.25	0.25	0.63	190.3
17	996.84	-8.81	264.59	-9.66	93.5	3.13	2.93	0.20	1.83	2.94	1312.18	0.18	0.63	167.2
23	996.99	-9.05	264.34	-10.02	92.6	3.07	2.85	0.23	1.78	2.85	1313.61	0.10	0.38	240.0
29	997.46	-9.63	263.72	-10.65	92.2	2.94	2.71	0.23	1.69	2.71	1317.19	0.40	0.88	157.0

$$[996.5 \ -8.05 \ 265.38 \ -8.78 \ldots 192.7]$$

- 입력 데이터의 각 x_t는 **벡터** 형태
- 시계열 데이터의 경우 **각 데이터를 이루는 Feature 값들을 원소**로 하여 벡터로 변환

자연어 데이터의 벡터 변환

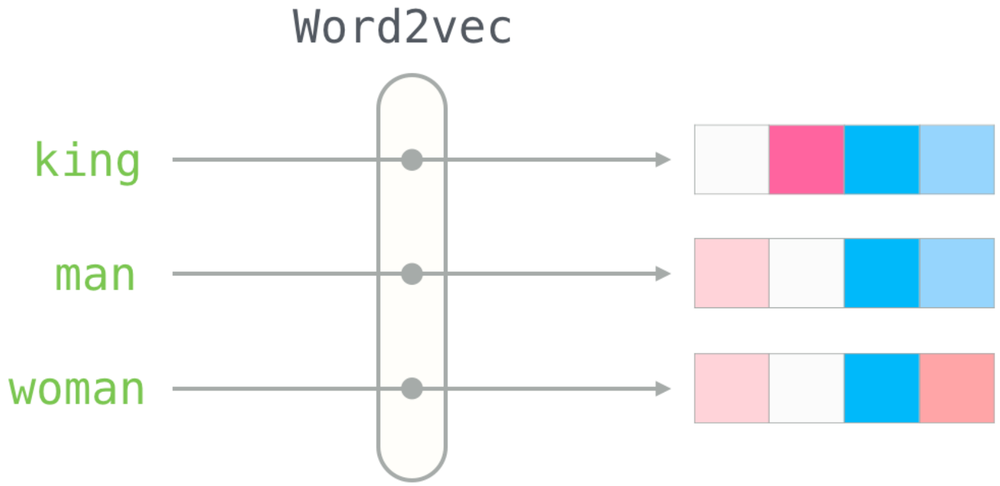
나는 ⟶ $[1\ 0\ 0\ 0]$ x_1

배가 ⟶ $[0\ 1\ 0\ 0]$ x_2

너무 ⟶ $[0\ 0\ 1\ 0]$ x_3

고프다 ⟶ $[0\ 0\ 0\ 1]$ x_4

One-hot Encoding

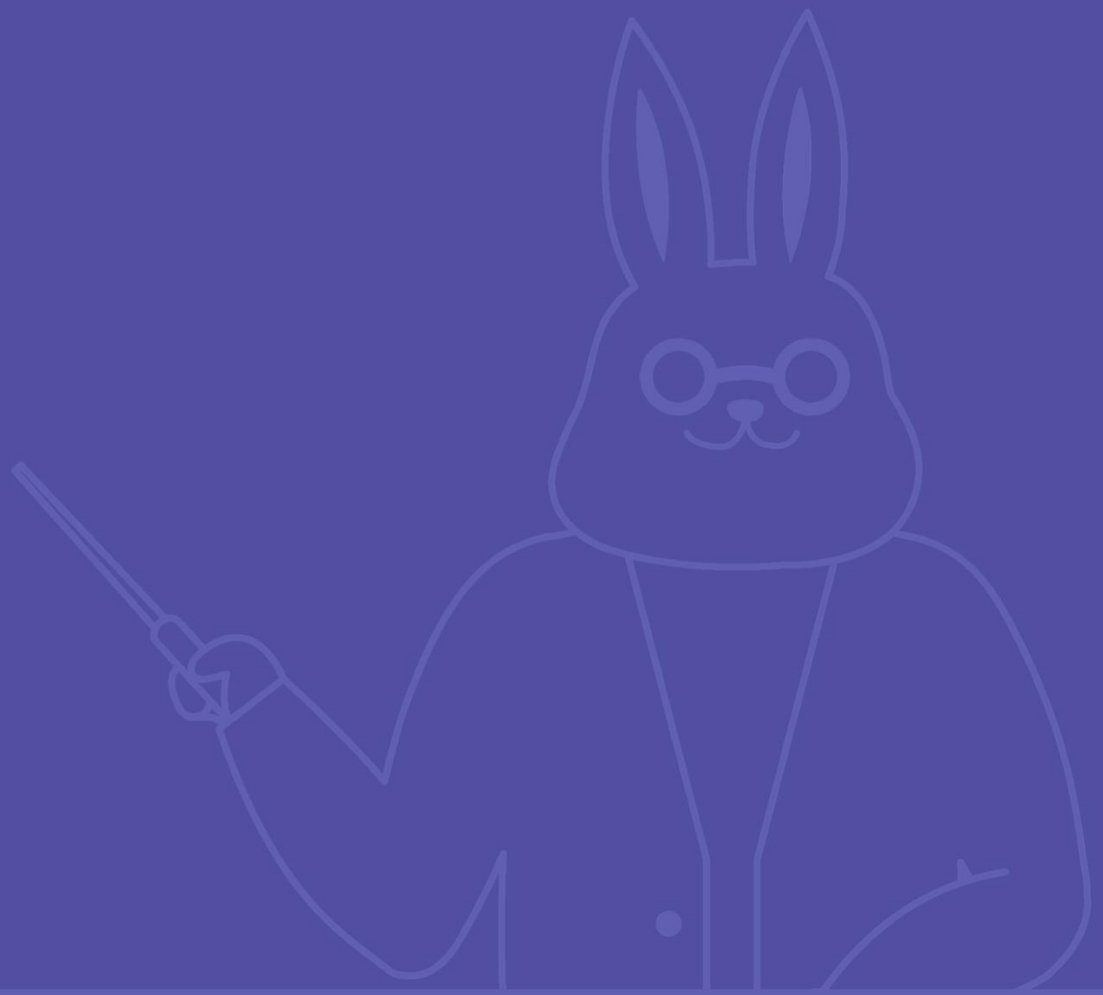


Word2Vec

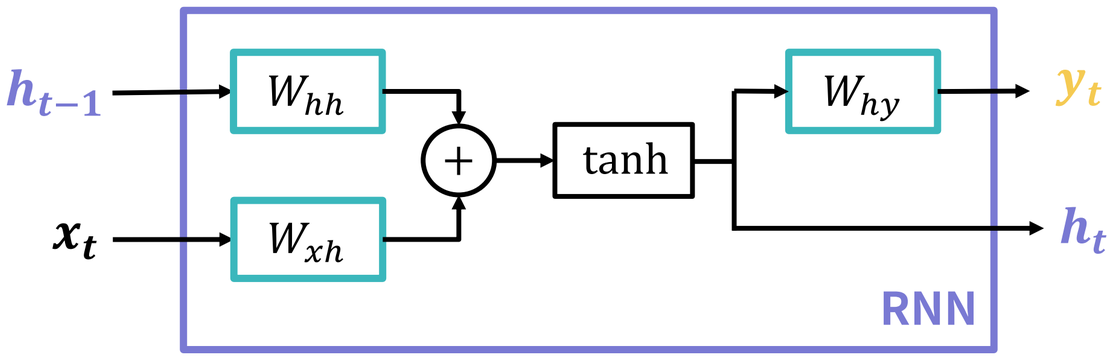
- **임베딩(Embedding)**: 각 단어들을 숫자로 이루어진 벡터로 변환
- 대표적인 임베딩 기법
  - **One-hot Encoding**
  - **Word2Vec**: 주어진 단어들을 벡터로 변환하는 기계학습 모델

04

Vanilla RNN

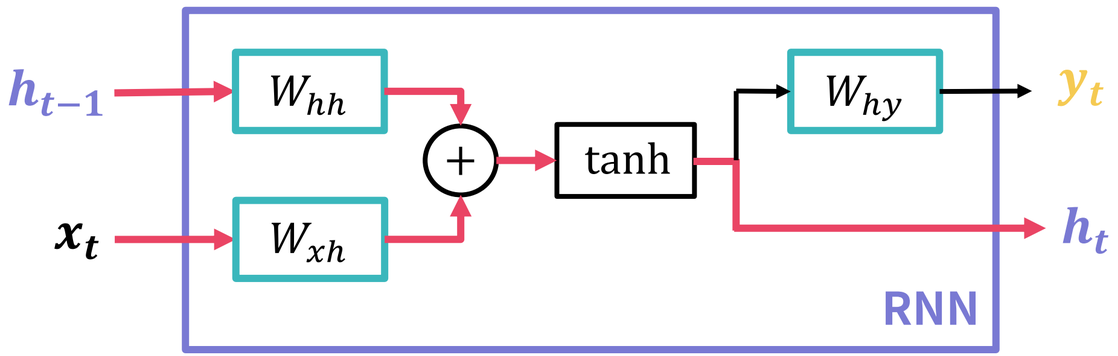


Vanilla RNN 소개

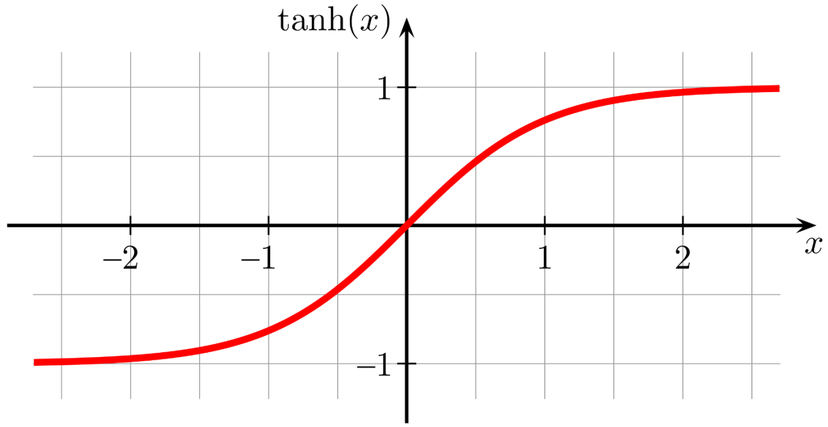


- 가장 **간단한 형태의 RNN** 모델: Simple RNN이라고도 불림
- 내부에 **세개의 FC Layer**로 구성
  - W_{hh}: **hidden state(h_{t-1})**를 변환하는 Layer의 가중치 행렬
  - W_{xh}: 한 시점의 **입력값 (x_t)** 을 변환하는 Layer의 가중치 행렬
  - W_{hy}: 한 시점의 **출력값 (y_t)** 을 변환하는 Layer의 가중치 행렬

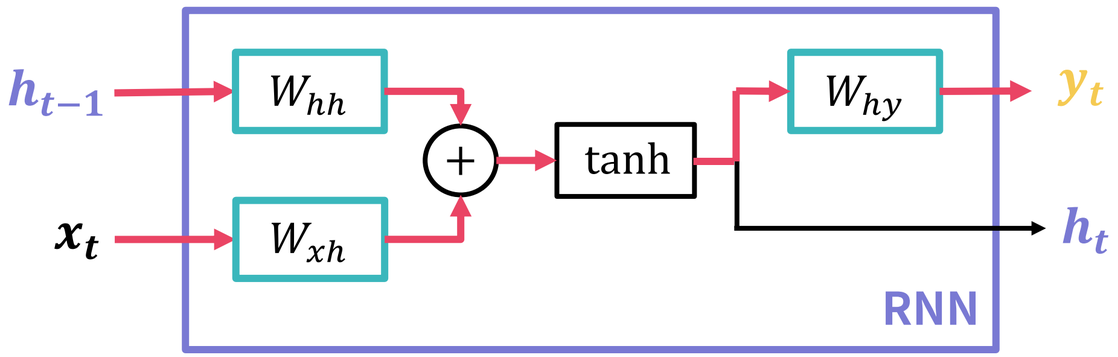
Vanilla RNN 연산 과정



- 현재 입력값(x_t)에 대한 **새로운 hidden state** 계산
- $h_t = \tanh(h_{t-1}W_{hh} + x_tW_{xh})$
  - tanh는 tangent hyperbolic 함수

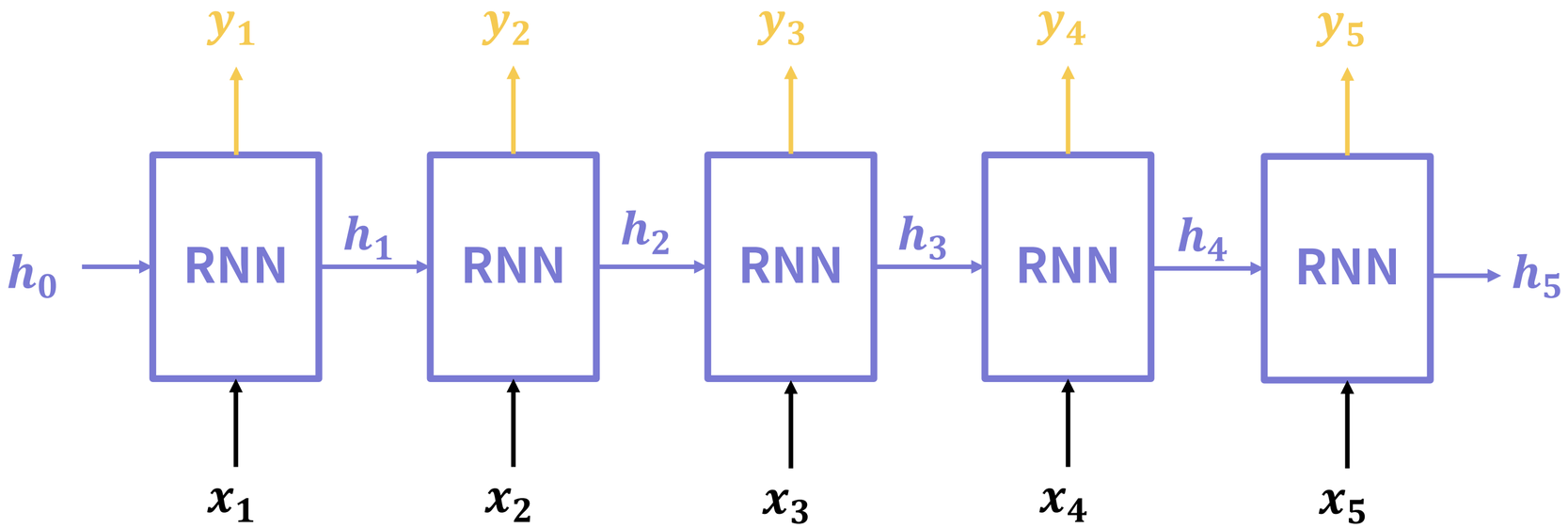


Vanilla RNN 연산 과정



- 현재 입력값(x_t)에 대한 새로운 **출력값** 계산
- $y_t = W_{hy}h_t$
  - 앞서 계산한 hidden state를 이용

시간순으로 보는 Vanilla RNN의 연산 과정



- 모델에 들어오는 각 시점의 데이터 x_t마다 앞서 설명한 연산 과정을 수행
- 입력값에 따라 반복해서 출력값과 hidden state를 계산
- **이전 시점에 생성된 hidden state를 다음 시점에 사용**

Vanilla RNN 의의

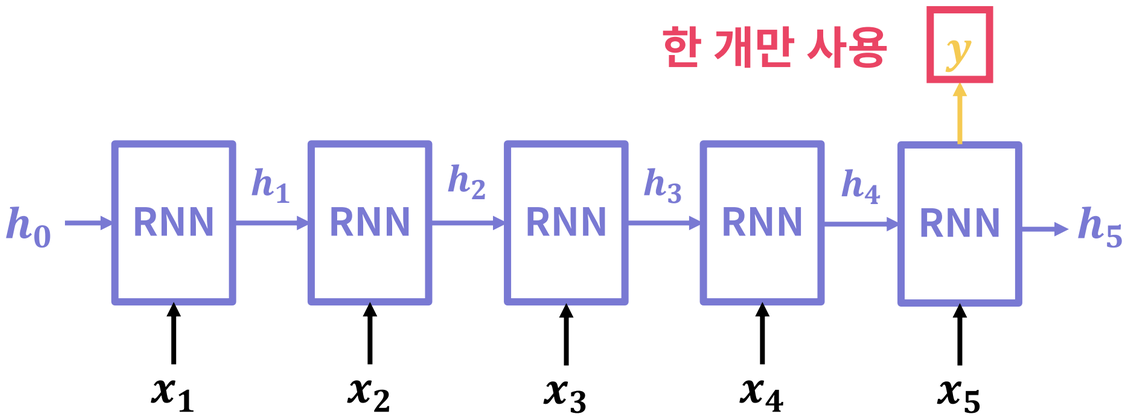
Hidden state의 의미

- 특정 시점 t까지 들어온 입력값들의 **상관관계**나 **경향성 정보**를 압축해서 저장
- 모델이 내부적으로 계속 가지는 값이므로 일종의 **메모리(Memory)**로 볼 수 있음
- 컴퓨터의 메모리와 일맥상통

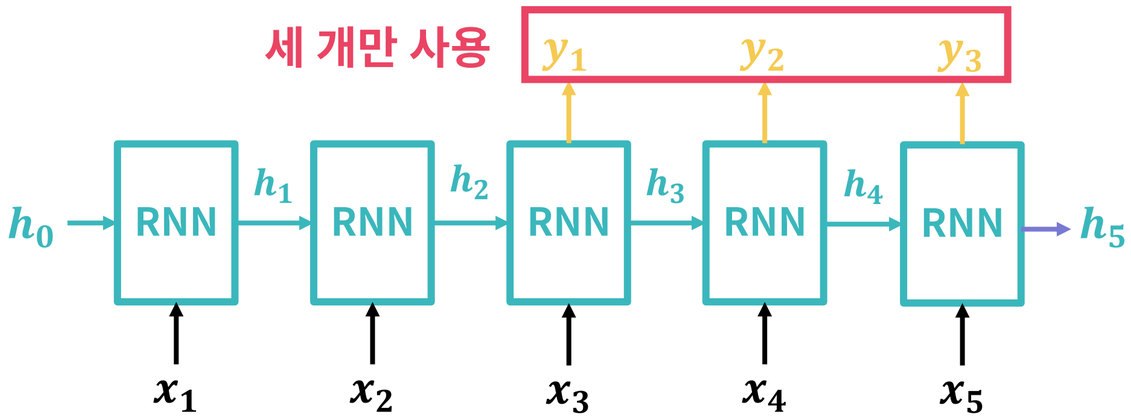
Parameter Sharing

- Hidden state와 출력값 계산을 위한 FC Layer를 모든 시점의 입력값이 **재사용**
- FC Layer 세 개가 **모델 파라미터의 전부**

Vanilla RNN의 종류



다대일 구조 (many-to-one)

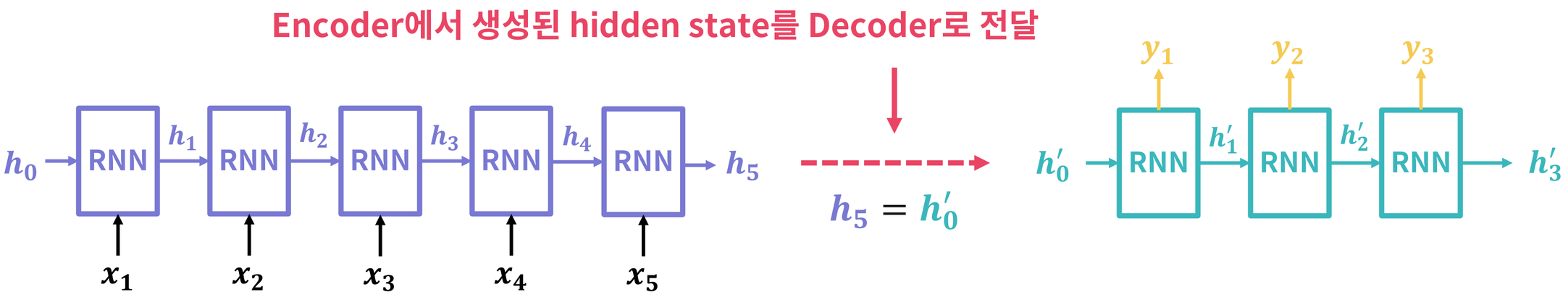


다대다 구조 (many-to-many)

사용할 입력값과 출력값의 구성에 따라 여러 종류의 RNN이 존재

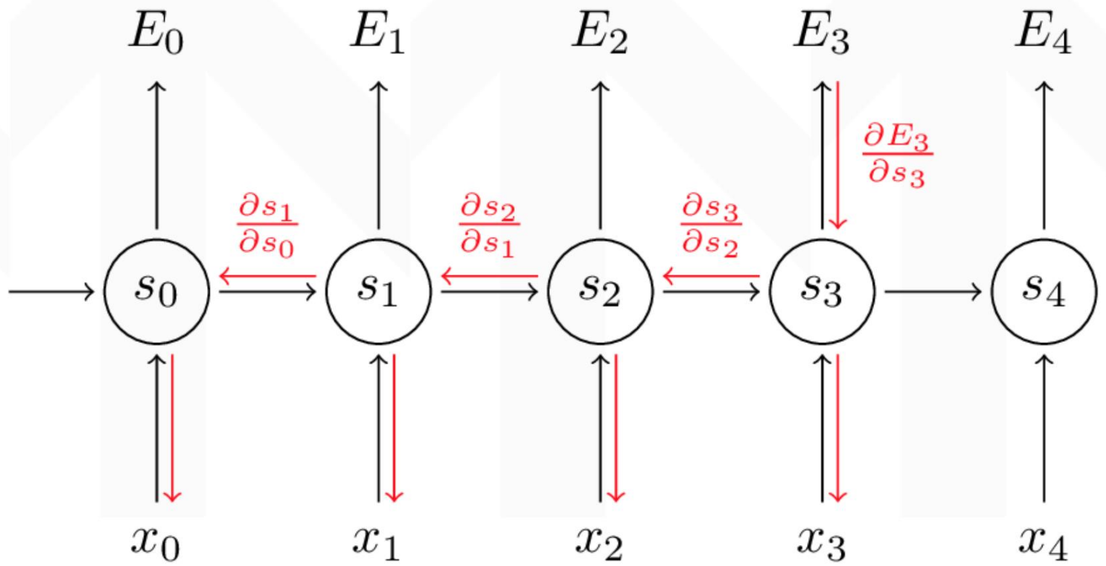
- **다대일(many-to-one):** 출력값을 한 시점의 값만 사용
- **다대다(many-to-many)**: 여러 시점의 입력값과 여러 시점의 출력값을 사용
  - 입력값과 출력값에 사용하는 **시점의 개수는 같을 수도 있고 다를 수도 있음**

Vanilla RNN의 종류



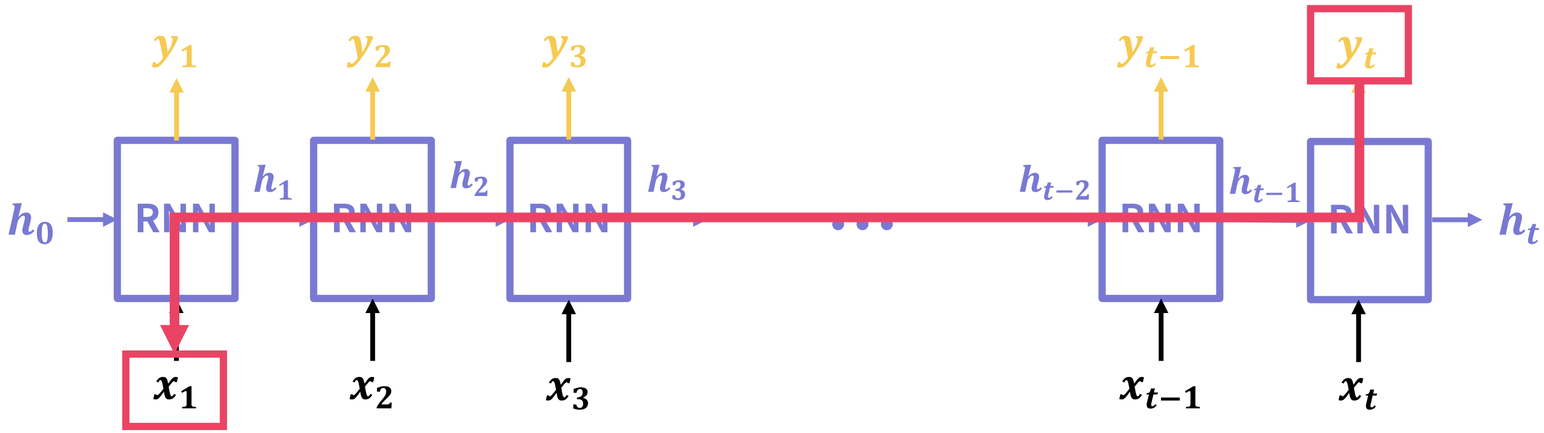
인코더-디코더(Encoder-Decoder): 입력값들을 받아 특정 hidden state로 인코딩한 후, 이 hidden state로 새로운 출력값을 만드는 구조

Vanilla RNN의 문제점



- RNN은 출력값이 시간 순서에 따라 생성
- 각 시점의 출력값과 실제값을 비교하여 손실(Loss)값을 계산
- 역전파 알고리즘이 시간에 따라 작동 → **Back-propagation Through Time (BPTT)**

Vanilla RNN의 문제점



- **입력값의 길이가 매우 길어질 경우**
  - 초기 입력값과 나중 출력값 사이에 전파되는 기울기 값이 매우 작아질 가능성이 높음
- **기울기 소실(Vanishing Gradient)** 문제가 발생하기 쉬움
  - 다른 말로 **장기 의존성(Long-term Dependency)**을 다루기가 어려움

크레딧

/* elice */

코스 매니저
김창환

콘텐츠 제작자
김창환

강사
김창환

감수자
-

디자이너
강혜정

연락처

TEL

070-4633-2015

WEB

https://elice.io

E-MAIL

contact@elice.io