

# Basic DeepLearning for NLP

[https://github.com/airobotlab/lecture\\_NLP\\_basic](https://github.com/airobotlab/lecture_NLP_basic)

# Contents

---

- 1) NLP Intro
- 2) Text Preprocessing
- 3) Traditional NLP
- 4) Clustering
- 6) Word2Vector
- 7) Subword tokenization
- 8) pytorch 101
- 9) Review of Deep Learning (CNN, RNN)
- 10) Search system

# Contents

---

- 11) Encoder-Decoder
- 12) Attention
- 13) Language Model
- 14) Transformer
- 15) NLP Tools (Hugging Face)
- 16) Transfer Learning&Downstream Task
- 17) Text Classification
- 18) Named Entity Recognition
- 19) Sequence Generation
- 20) Domain specific Language Model

# Contents

---

- 21) Zero-shot Learning (GPT-3)
- 22) Speech-to-Text
- 23) Text-to-Speech
- 24) OCR
- 25) NLP product deploy (NVIDIA TensorRTTriton)

# 1) AI가 중요하다는데?



AI는 인류 역사상 최대 수준의 혁명을 초래할 것입니다

**Let's be honest before we start.**

---

**진짜 AI를 적용하고 싶으신가요?**

**AI로 기존과 다른 혁신적인 무언가를 만들어보자**

**or**

**AI를 했다고 하고 싶으신가요??**

**그냥 내부 경쟁, 과제를 따기 위한 키워드, 대외 홍보 수단**

Let's be honest before we start.

AI부서가 진정 AI로 획기적인것을 만들길 바라시나요?

or

남들 다 하니까 그냥 몇 명한테 한번 해봐

“역시 안되잖아?” 하며 다시 관성대로 돌아가고 싶으신가요?



**Let's be honest before we start.**

---

**Recent Deep Learning?**

**데이터가 엄청 많고 복잡한 문제**

**or**

**Traditional Machine Learning?**

**데이터 수가 적고, 복잡하지 않은 문제**

## 1) AI가 중요하다는데?

---

AI 기술 도입은 선택이 아닌 필수

동의 하시나요?

# 1) AI가 중요하다는데?

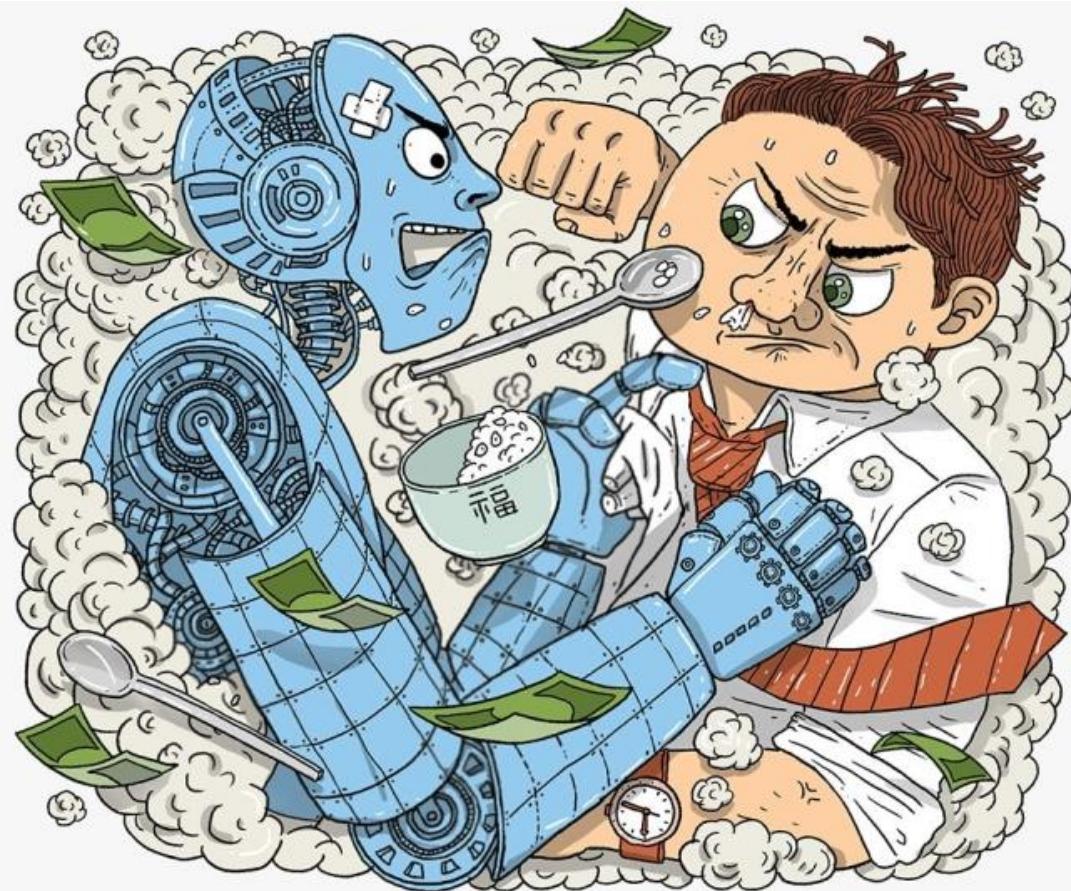
---

모두가 AI를 하지만

제대로 하는 곳은 찾기 힘듭니다

## 2) AI 투자 실패 1

형량예측부터  
변호사상담까지  
**로톡**



### 스타트업 vs 전통 사업자 갈등 사례

■ 스타트업 ■ 전통 사업자

AI 기반  
사업

**삼쩜삼** AI 기반 세금신고 및 환급 서비스  
**세무사고시회** “불법 세무 대리”

증개  
사업

**빅밸류** AI 기반 부동산 시세 자동 산정 서비스  
**감정평가사협회** “무자격 감정 평가”

로톡  
법률 서비스 플랫폼

**대한변협** “변호사법상 금지된 증개·알선 행위”

**강남언니** 미용·의료 정보 플랫폼

**의사협회** “불법 의료 광고”

**다원증개** 반값 증개 서비스 플랫폼

**공인증개사협회** “불법 증개 행위”

## 2) AI 투자 실패 2

### 돈 못버는 골칫덩이됐다… AI 선구자 ‘왓슨’의 몰락

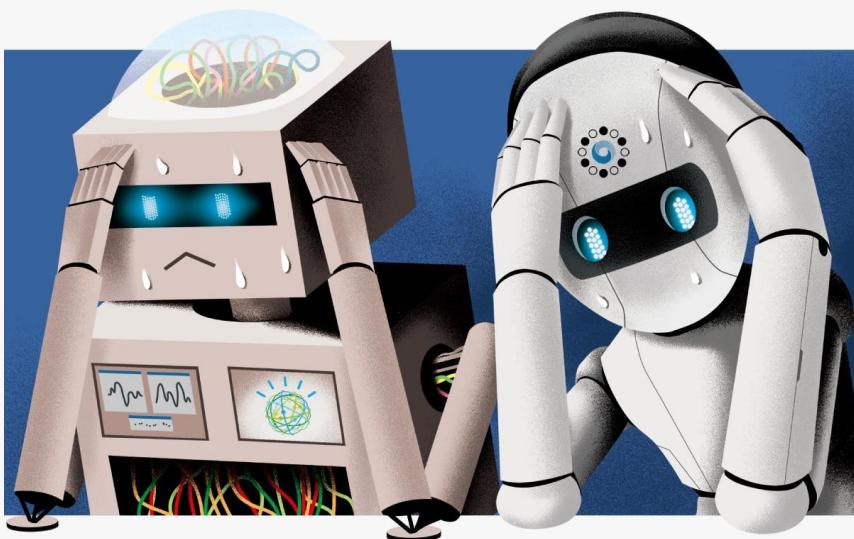
AI 시대 연 지 10년 만에 IBM 골칫덩이로

박건형 기자

입력 2021.07.19 20:15



2011년 2월, 미국 ABC방송 퀴즈쇼 제퍼디에 사상 처음으로 사람이 아닌 우승자가 등장했다. 두 명의 인간 챔피언을 압도적으로 누른 주인공은 IBM의 인공지능(AI) 수퍼컴퓨터 ‘왓슨(Watson)’이었다. 사람보다 훨씬 문제를 잘 이해하고 빠르게 답을 제시하는 왓슨에 전세계는 ‘AI 시대’가 도래했다며 흥분했다. 다음날 IBM은 “우리는 왓슨을 의료, 금융, 법률, 학술 등 다양한 분야에 적용할 방법을 찾고 있다”고 대대적으로 홍보했다. 하지만 10년이 지난 지금 왓슨은 IBM의 골칫덩이로 전락했다. 왓슨 사업은 대부분 중단됐고 IBM은 왓슨 의료 사업부 매각을 추진하고 있다. 미국 뉴욕타임스(NYT)는 17일(현지 시각) “왓슨의 원대한 비전은 사라졌고 AI에 대한 과장과 오만함을 일깨우는 사례가 됐다”고 보도했다. 한 때 AI 혁명의 선두 주자이자 컴퓨터 사업을 대체할 IBM의 강력한 무기로 주목받았던 왓슨은 왜 실패했을까.



갈 길 잃은 인공지능 선구자들

왓슨(IBM)	딥마인드 알파고(구글)
2006년	개발 시작
수퍼컴퓨터 기반의 방대한 지식 데이터베이스	방식
퀴즈쇼 제퍼디에서 인간 챔피언 꺾고 우승	주요 성과
의료·금융·법률·학계에서 인간 전문가의 역할 대체	당초 목표
-암 진단 프로젝트 전면 중단 -왓슨 헬스 매각 추진	기후변화 대응과 생명 현상 규명
	현재 상태
	-영국 전력망 효율화 계획 포기 -에너지팀 해체

## 2) Watson의 실패 이유= '경영진의 잘못된 판단'

- 기술보다 마케팅 앞세운 왓슨의 실패
- 가능성을 지나치게 과대평가
- 완성되지도 않은 기술을 출시하는데 급급
- 당시 IBM의 최고 경영진은 대부분 마케팅 전문가, 기술 X
- Watson이 퀴즈쇼라는 제한된 환경에 맞춰 제작됐다는 걸 이해 X
- 마케팅과 홍보에 막대한 돈을 쏟아 부은 뒤 왓슨의 활용처를 찾기 시작
- '말보다 마차가 앞서가는 꼴'이라는 내부 비판은 묵살



## 2) Watson의 실패 이유= '경영진의 잘못된 판단'

- “암 데이터는 IBM 연구진의 생각보다 훨씬 복잡
- 오염된 데이터도 왓슨의 정확도를 높이는데 장애
- 왓슨은 의사가 쓴 메모 조차 제대로 인식 불가
- 암 정복이라는 원대한 구상이 현실화되지 않자 왓슨 프로젝트는 속속 중단
- 노스캐롤라이나대와 뉴욕 메모리얼 슬로언 케터링 병원 암센터는 암진단용 왓슨 개발을 중단
- 휴스턴 MD앤더슨 병원은 왓슨에 4년간 6200만달러(약 771억원)를 쏟아 부은 뒤 실패를 선언
- 왓슨 의료 사업부 매각을 추진하고 있지만 뚜렷한 구매자 없음
- 시장에서 왓슨은 수익성이 아주 낮거나 아예 없는 사업으로 평가 by 월스트리트저널
- 주가 10년간 10% 하락



## 2) Watson의 실패 이유= '경영진의 잘못된 판단'

- 반면 아마존·페이스북·마이크로소프트 같은 경쟁자들은 승승장구&주가 급등
- 음성인식 비서나 이미지 인식 같은 실질적인 제품이나 서비스
- IBM은 '암 정복'처럼 당장 성과를 기대할 수 없는 거대한 목표



## 2) 딥마인드의 실패 이유= '통제된 환경에서만 제대로'

---

- 딥마인드도 핵심 프로젝트 접어
- '알파고(AlphaGo)'를 개발한 구글 딥마인드 역시 갈팡질팡
- 바둑을 정복한 뒤 다음 목표로 '전력 효율화를 통한 기후변화 대응'
- 구글 데이터센터에 알파고를 투입해 전력을 40% 절감하는 성과
- '딥마인드 에너지'라는 별도 팀을 구성해 영국 국영 내셔널그리드와 함께 영국 국가 전력 사용량을 10% 이상 줄이겠다고 선언
- 하지만 이 프로젝트는 지난해 무산됐고 팀은 해체
- '딥마인드의 기술이 바둑이나 체스처럼 **통제된 환경에서만 제대로 작동**할 뿐, 현실 세계의 복잡성과 예측 불가능성에는 맞지 않는다' by CNBC

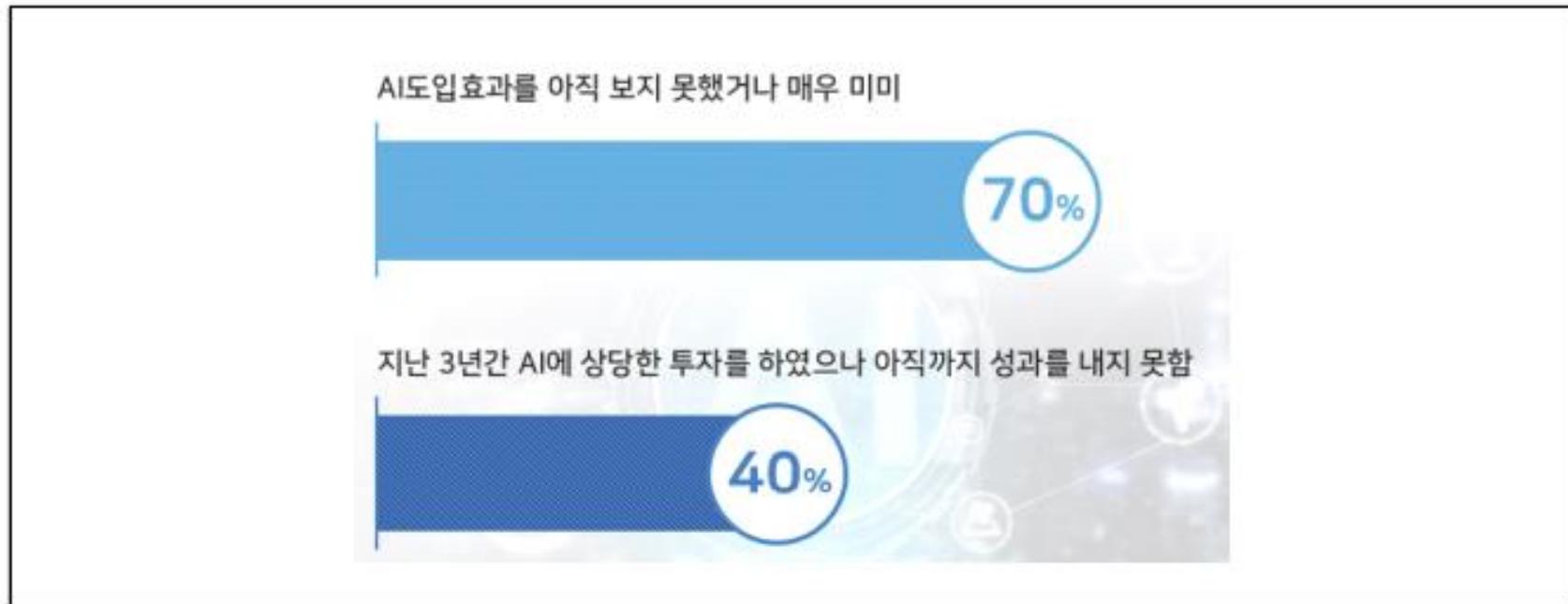
## 2) 딥마인드의 실패 이유= '통제된 환경에서만 제대로'

- 구글은 혁신 기술에 과감한 투자를 하는 것으로 유명
- 수익성이 떨어진다고 판단하면 가차없이 사업을 정리
- 보스턴 다이내믹스를 2013년 인수했다가 2017년 소프트뱅크에 매각
- 2014년 딥마인드를 인수한 뒤 2조원 이상을 투자한 구글이 언제까지 인내할지 모르는 상황



## 2) AI 투자 실패

[그림 3] AI 도입효과



주: 97개국 29개 산업에 종사하는 2,555명을 대상으로한 설문조사 결과

자료: MIT & BCG (2019.10)의 설문결과를 그래프로 그림

## 2) AI 투자 실패

---

- 2021, 정부 AI에 2.3조 투입
- But, 80% 이상의 인공지능 프로젝트는 실패 by Gartner, VentureBeat
- 산업 현장에 그대로 적용하기에는 무리

---

실패하지 않기 위해서는?

- 1) AI에 대한 이해 필수
- 2) AI의 한계 받아드리기
- 3) AI는 아주 일부일 뿐. MLOps 확충 필수
- 4) '유능한' CTO에게 맡기자
- 5) 인재영입. 최선을 다하라
- 6) 사업 부서간 협업 및 구조혁신 필수!!
- 7) 실패한다면? 100% C레벨 때문이다

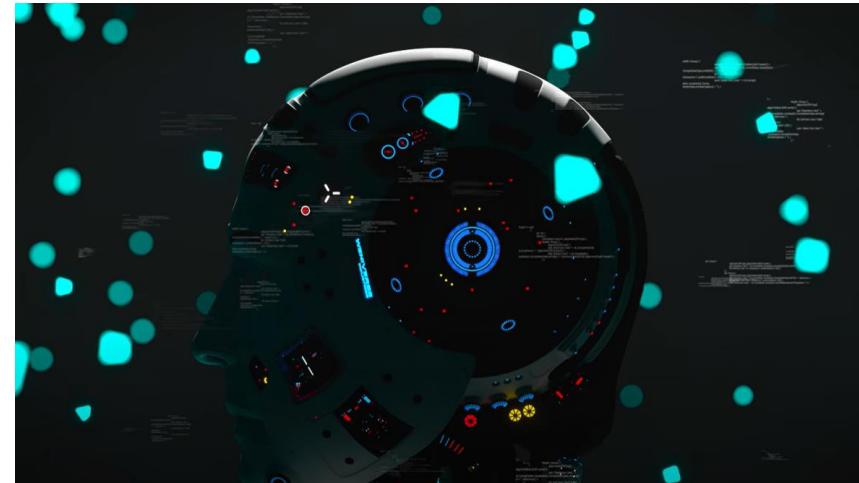
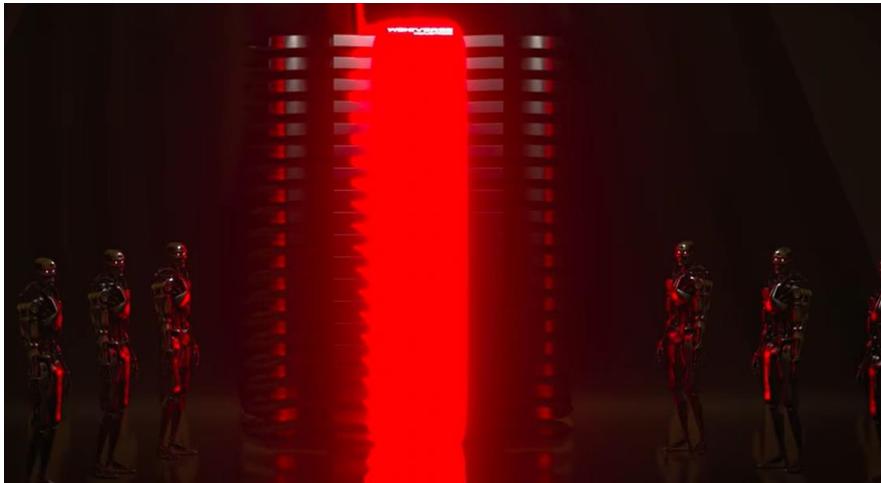


# C레벨이 공부를 안하면



# 1) AI에 대한 이해 필수

- AI공부를 해야만 한다..



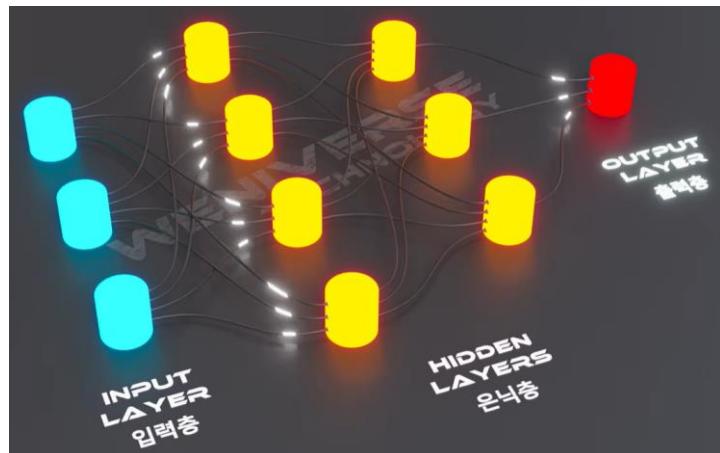
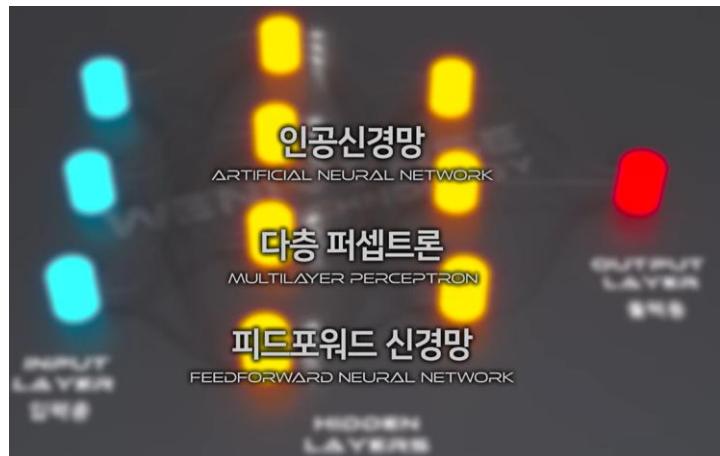
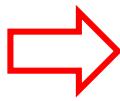
# 1) AI에 대한 이해 필수

- AI공부를 해야만 한다..
- Why??
  - AI로 할 수 있는것과 없는 것 파악



# 1) AI에 대한 이해 필수

- AI공부를 해야만 한다..
- Why??
  - AI로 할 수 있는것과 없는 것 파악

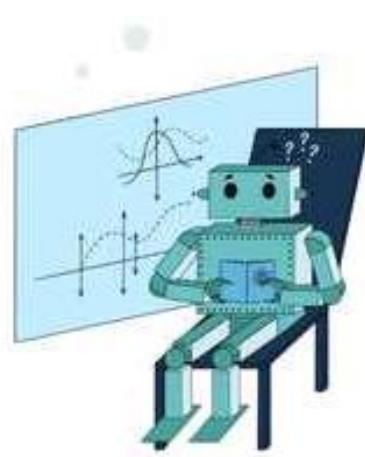


# 1) AI에 대한 이해 필수

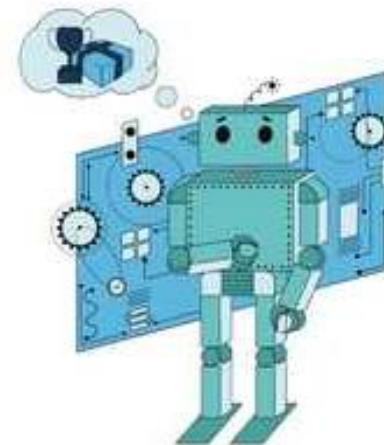
- AI공부를 해야만 한다..
- AI에 어떤 종류가 있는지?



Supervised Learning



Unsupervised Learning

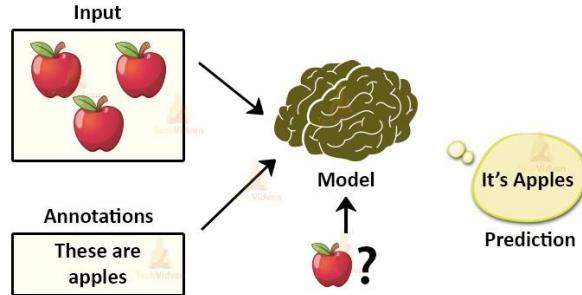


Reinforcement Learning

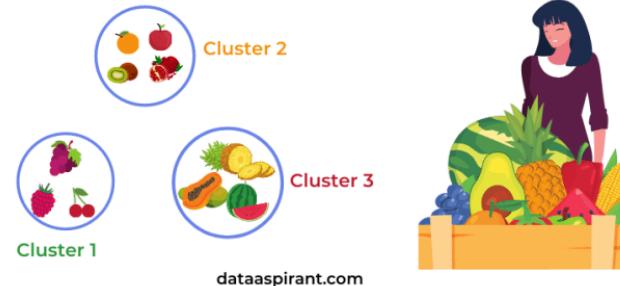
# 1) AI에 대한 이해 필수

- AI공부를 해야만 한다..
- AI에 어떤 종류가 있는지?
- 각각은 무엇인지?

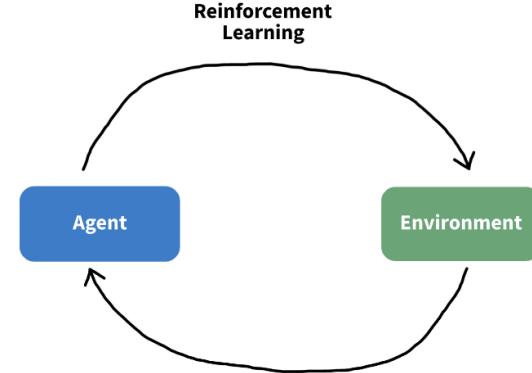
## Supervised Learning in ML



## Unsupervised Learning

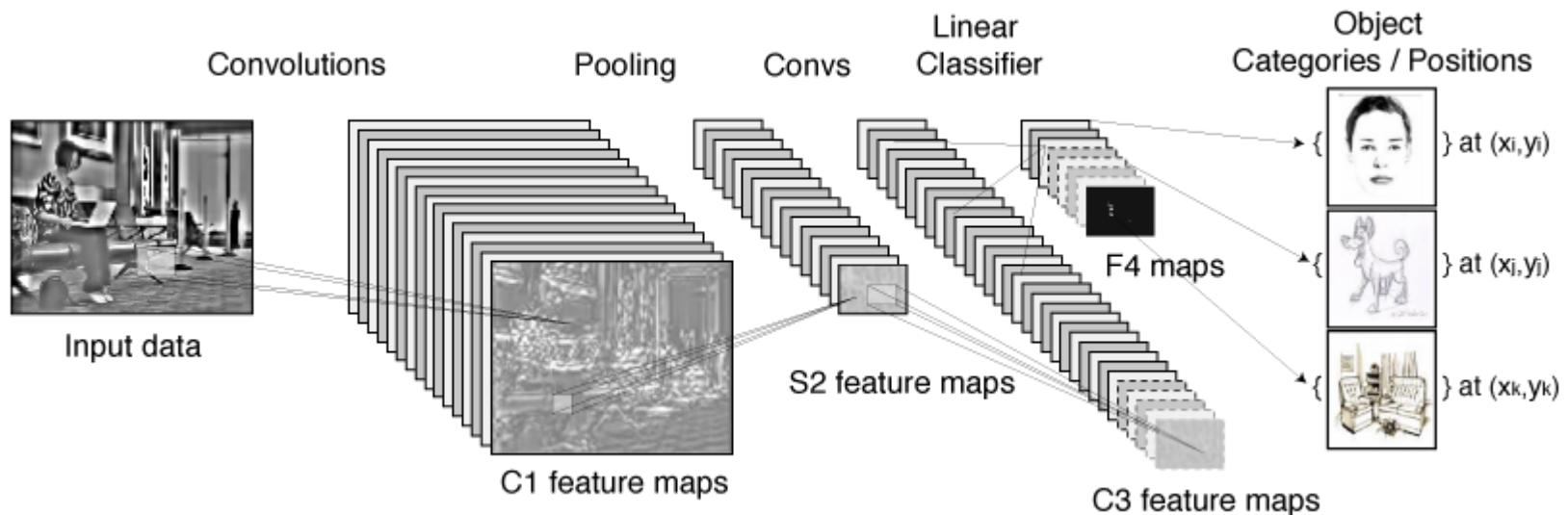


## Reinforcement Learning

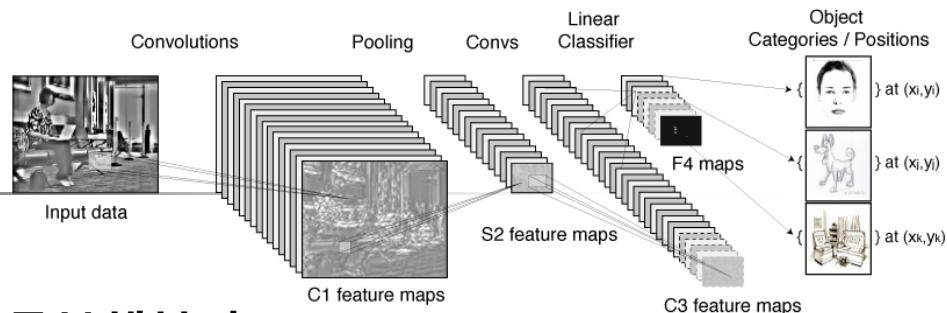


# 1) AI에 대한 이해 필수

- AI공부를 해야만 한다..
- Why??
  - AI로 할 수 있는것과 없는 것 파악



# 1) AI에 대한 이해 필수



- 대체 AI가 뭔데? 학원&책한권 사서 공부해볼까?
- 어? 이게 끝이야? 뭐 이렇게 쉬워?

```
import tensorflow as tf
mnist = tf.keras.datasets.mnist
```

```
(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0
```

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(10, activation='softmax')
])
```

```
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

```
model.fit(x_train, y_train, epochs=5)
```

```
model.evaluate(x_test, y_test)
```

라이브러리  
keras 데이터베이스

튜플 데이터 준비  
표준화

모델 네트워크 정의

optimizer, loss,  
metrics 설정

학습(training)

테스트(test)

```
import tensorflow as tf
mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0
```

라이브러리  
keras 데이터베이스  
튜플 데이터 준비  
표준화

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(10, activation='softmax')
])
```

모델 네트워크 정의

```
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

optimizer, loss,  
metrics 설정

```
model.fit(x_train, y_train, epochs=5)
```

학습(training)

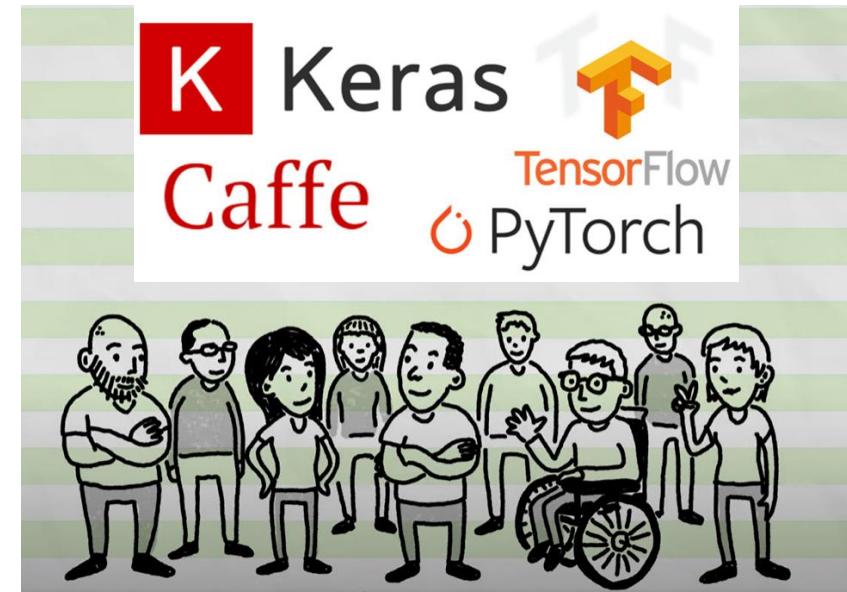
```
model.evaluate(x_test, y_test)
```

테스트(test)

오 이거 별거 아닌데?  
몇 줄 툭툭 치고 탁 누르면  
Tensorflow라는게 파바박 팍 해서  
인공지능을 촉! 하고 만들어주는데??



# Tensorflow/Keras/Pytorch는 인공지능 개발을 위한 '언어'일 뿐



# 실패 이유는 뭘까?

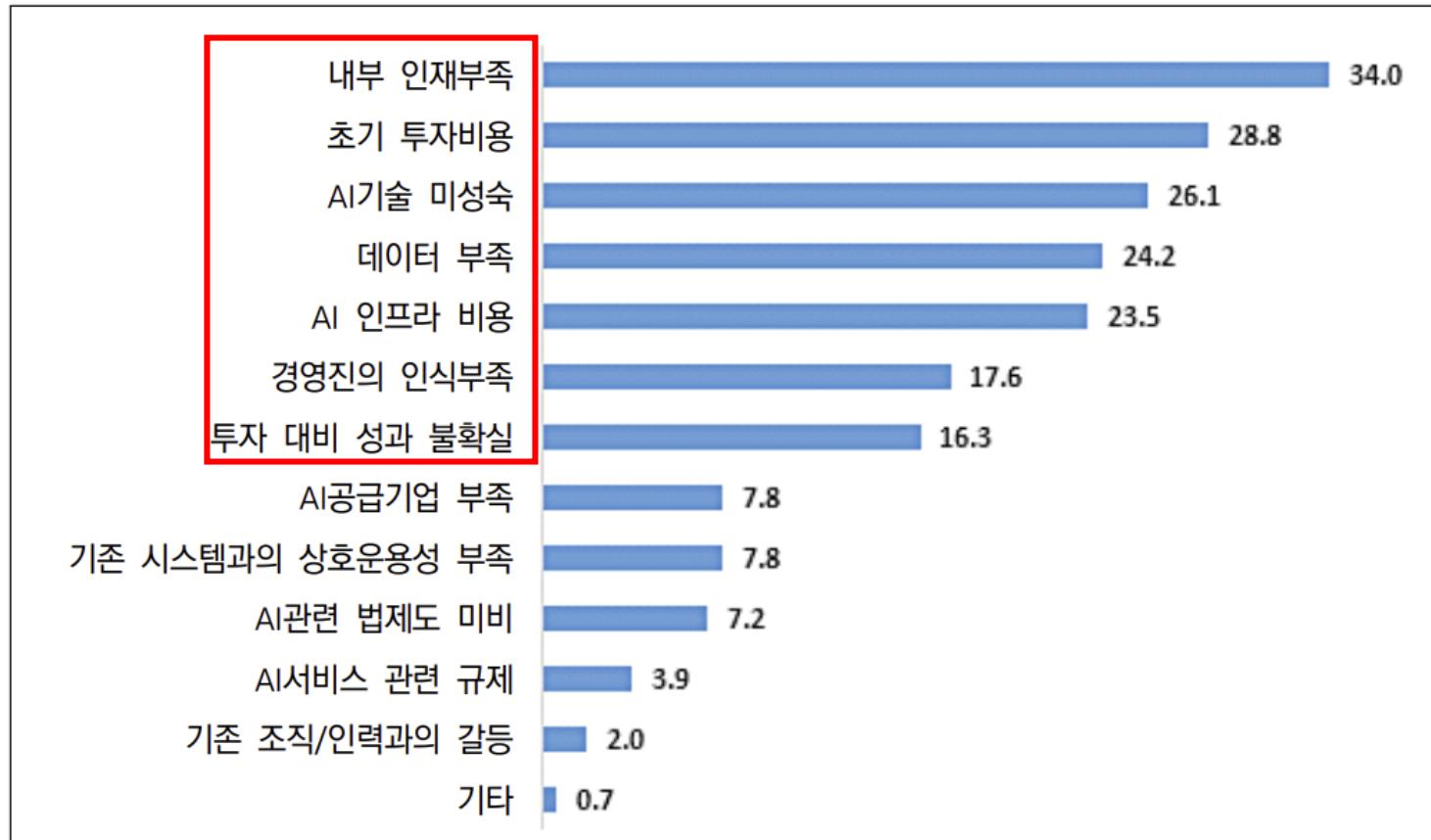
## 인력? 기술?

## AI의 이해



## 2) AI 도입 장애물

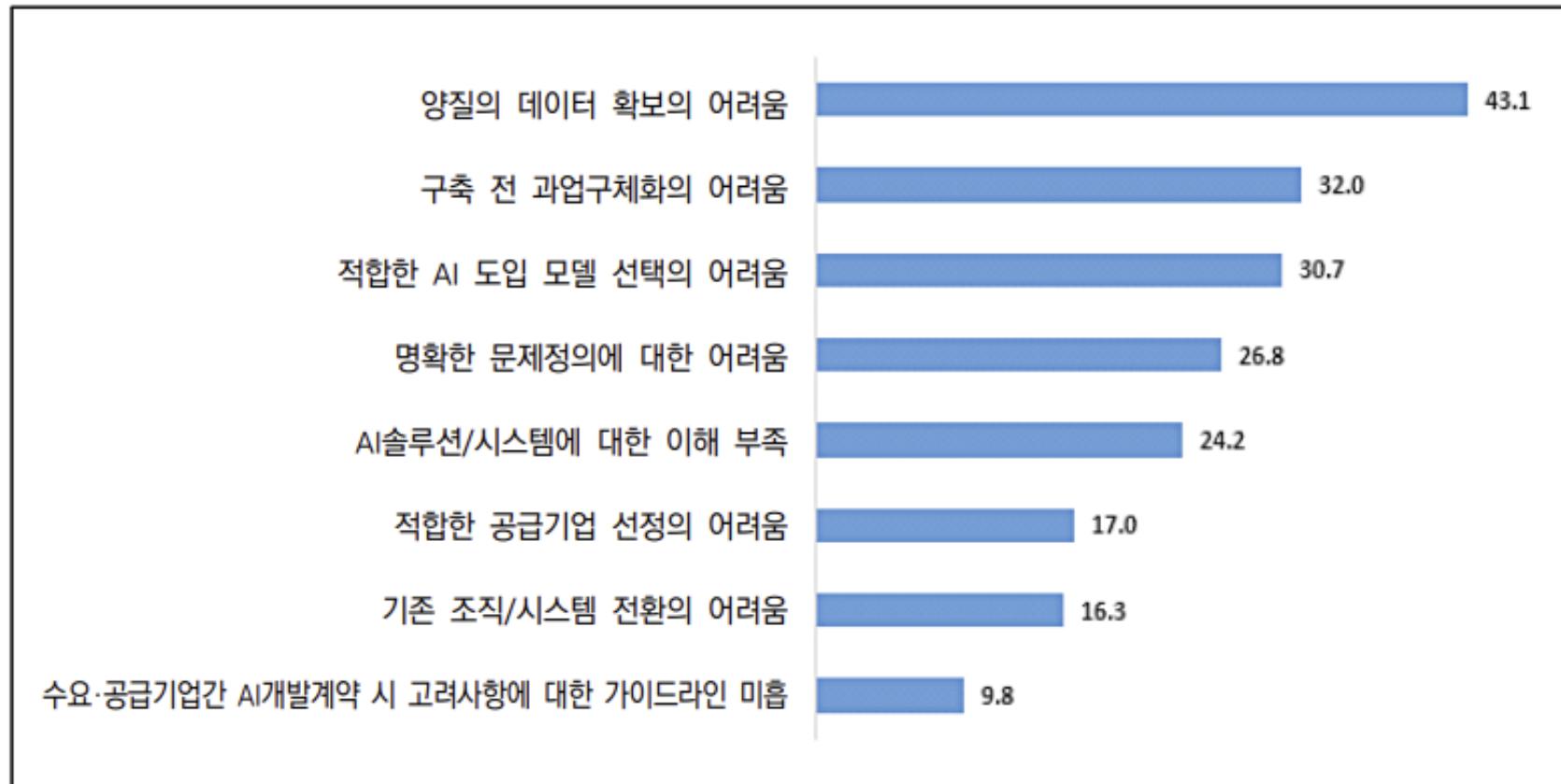
[그림 9] AI 도입의 장애물 (1+2 순위 응답) (단위: %)



주: AI 수요·공급기업 152개를 대상으로 AI 도입의 장애물에 대한 설문조사 수행 결과 1순위와 2순위 응답을 합친 결과로 응답의 총합은 200%

## 2) AI 도입 장애물

[그림 10] AI 도입 사전준비단계의 저해요인 (1 + 2 순위 응답) (단위: %)

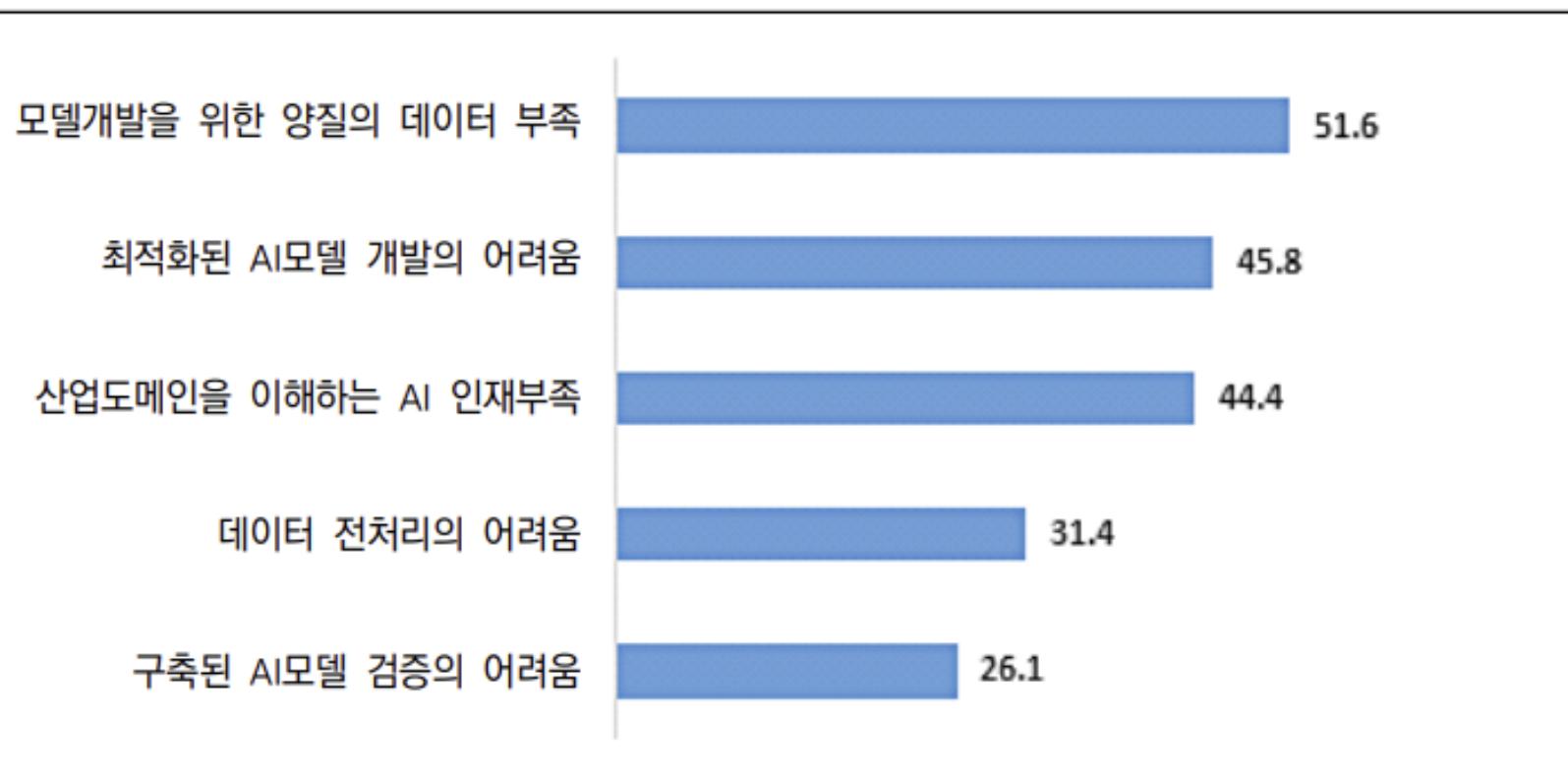


주: 국내 152개 AI 수요·공급기업 설문조사 결과

1순위와 2순위 응답을 합친 결과로 응답의 총합은 200%

## 2) AI 도입 장애물

[그림 11] AI 모델 개발 단계의 저해요인 (1+2 순위 응답) (단위: %)

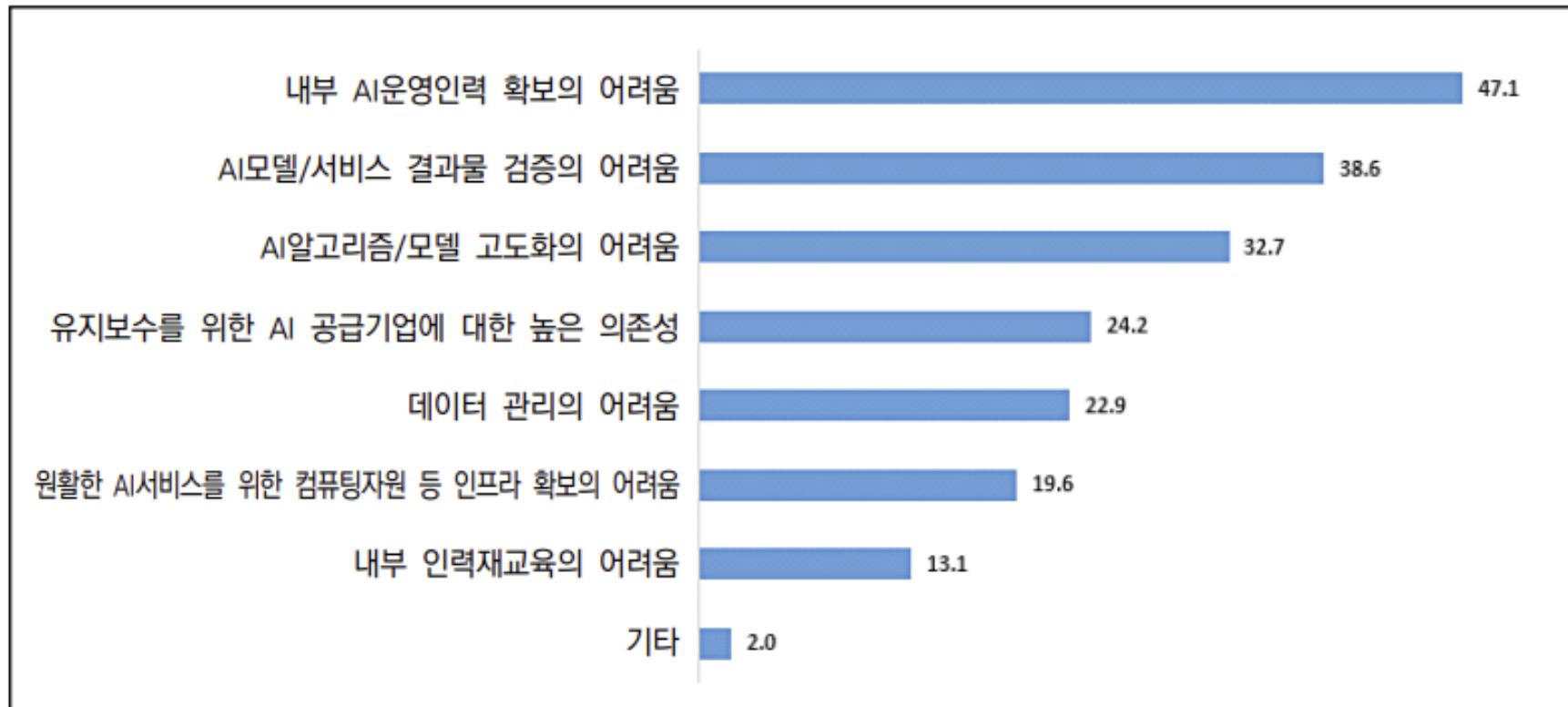


주: 국내 152개 AI 수요·공급기업 설문조사 결과

1순위와 2순위 응답을 합친 결과로 응답의 총합은 200%

## 2) AI 도입 장애물

[그림 12] AI 서비스 운영 단계의 저해요인 (1 + 2 순위 응답) (단위: %)



주: 국내 152개 AI 수요·공급기업 설문조사 결과

1 순위와 2순위 응답을 합친 결과로 응답의 총합은 200%

실패하는 흔한 프로젝트가 되지 않기 위해선??

필수 AI 프로세스

## 2) AI의 한계 받아드리기 - data

## ▪ 2.1) 보유한 데이터는?

- 데이터는 어떤 종류인가?

- ## ■ 이미지?

- #### ■ 음성?

- ## ■ 텍스트?

- Label은 어떤 종류인가? (이미지)

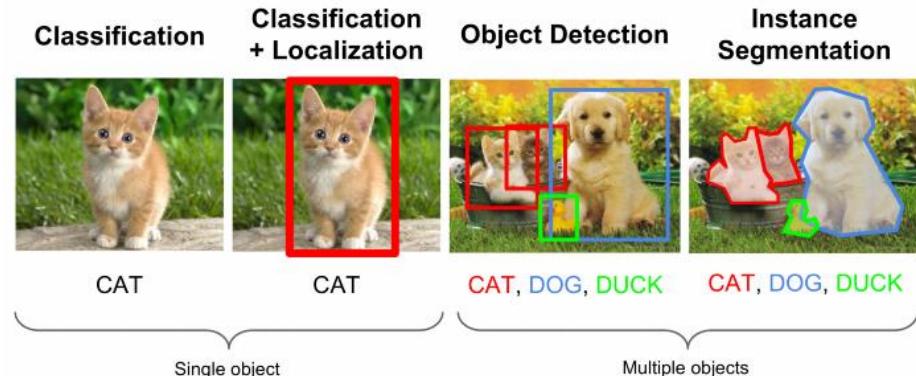
- #### ▪ 분류?

- ## ■ 회기?

- ## ▪ Object detection?

- ## ▪ Segmentation?

- ## ▪ OCR?



## 2) AI의 한계 받아드리기 - data

- 2.1) 보유한 데이터는?

- 데이터는 어떤 종류인가?

- 이미지?

- 음성?

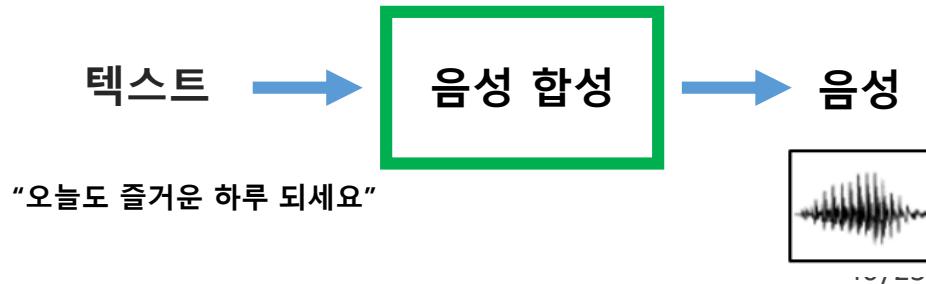
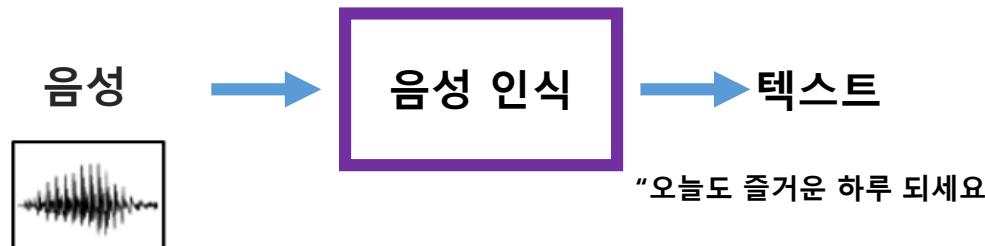
- 텍스트?

- Label은 어떤 종류인가? (음성)

- 음성 분류?

- 음성 인식?

- 음성 합성?



## 2) AI의 한계 받아드리기 - data

---

- 2.1) 보유한 데이터는?
  - 데이터는 어떤 종류인가?
    - 이미지?
    - 음성?
    - **텍스트?**
  - Label은 어떤 종류인가? (텍스트)
    - 분류?
    - 번역/요약/챗봇?
    - 생성?

## 2) AI의 한계 받아드리기 - data

- 2.1) 보유한 데이터는?
  - 데이터가 충분한가?



## 2) AI의 한계 받아드리기 - data

- 2.1) 보유한 데이터는?

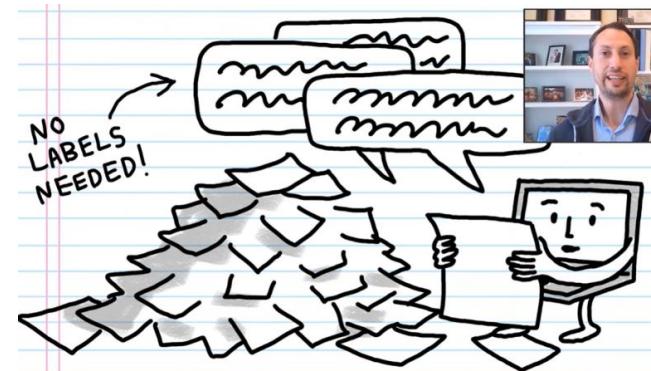
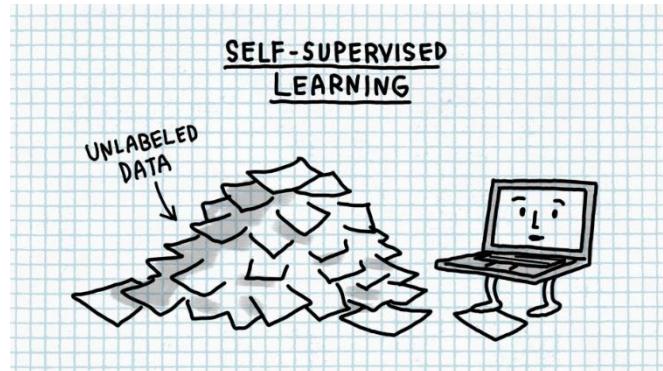
- 데이터가 충분한가?

- 서버에 가득 쌓여는 있다. 한 1000개?

- **PDF문서는 데이터가 아니다!!** 파싱이 되어있는가? 안됐다면 없는 거다!

- Label이 있는가?

- 없다. 그 데이터는 레이블이 없는 데이터다. 없다!



## 2) AI의 한계 받아드리기 - data



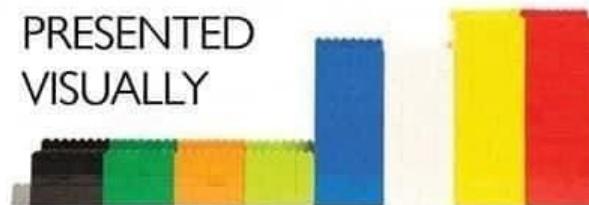
SORTED



ARRANGED



PRESENTED  
VISUALLY



EXPLAINED  
WITH A STORY



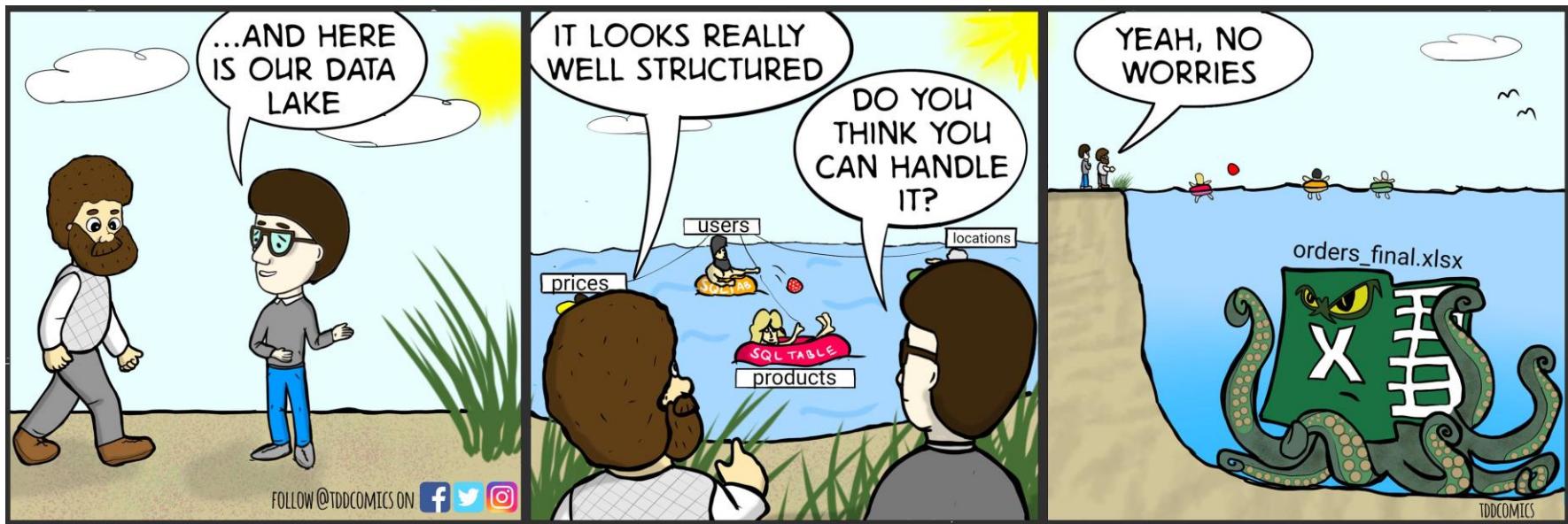
## 2) AI의 한계 받아드리기 - d

### ▪ 2.1) 보유한 데이터는?

- 데이터가 충분한가?

- 서버에 가득 쌓여는 있다. 한 1000개?

- **PDF문서는 데이터가 아니다!!** 파싱이 되어있는가? 안됐다면 없는 거다!



## 2) AI의 한계 받아드리기 - d

### ▪ 2.1) 보유한 데이터는?

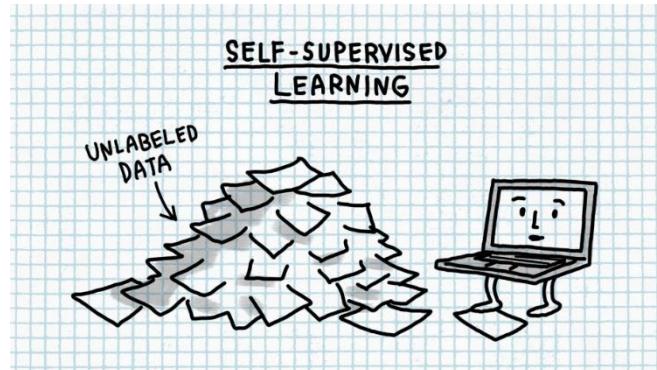
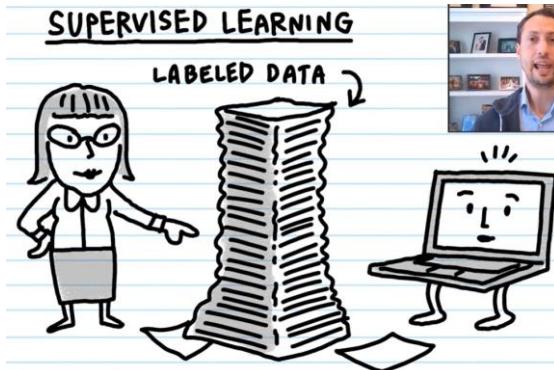
- 데이터가 충분한가?

- 서버에 가득 쌓여는 있다. 한 1000개?

- **PDF문서는 데이터가 아니다!!** 파싱이 되어있는가? 안됐다면 없는 거다!

- Label이 있는가?

- 없다. 그 데이터는 레이블이 없는 데이터다. 없다!



## 2) AI의 한계 받아드리기 - data

- 2.1) 보유한 데이터는?
  - 데이터가 충분한가?



## 2) AI의 한계 받아드리기 - data

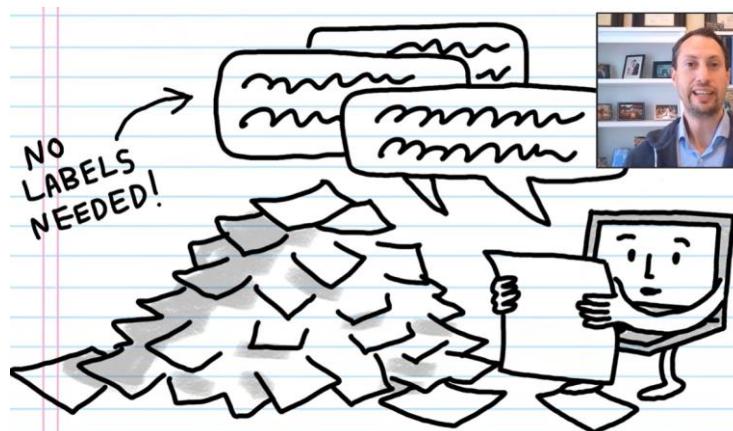
- 2.1) 보유한 데이터는?

- Label이 있는가?

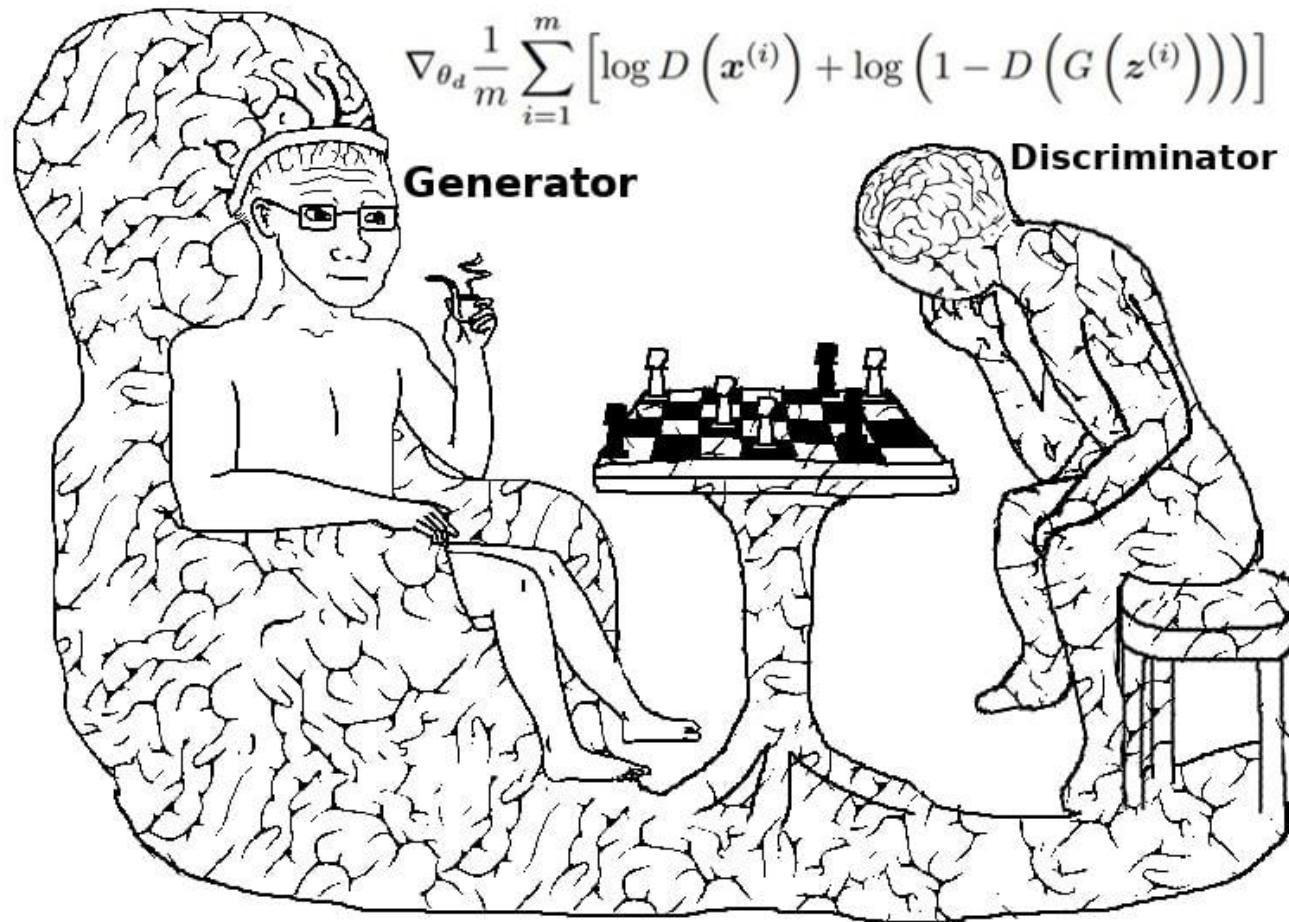
- 없다. 그 데이터는 레이블이 없는 데이터다. 없다!

- Label이 없어도 학습할 수 있는 Semi/Self-Supervised learning이 있다는데??

- Not possible for application



## 2) AI의 한계 받아드리기 - data



## 2) AI의 한계 받아드리기 - data

clearbox.ai

- What the hell is this?

Generative model

Synthetic  
data

Original  
dataset

## 2) AI의 한계 받아드리기 - data

---

### ▪ 1. 학습데이터 부족

- 지도학습이 가장 강력한 성능
- 지도학습의 데이터는 입력 데이터와 출력 데이터의 쌍으로 구성
- 학습 데이터의 **양**이 인공지능 성능에 큰 영향
- 하지만 학습 데이터 **구축이 쉽지 않음**
- 의료 영상 이미지 학습 데이터를 구축
  - 고도로 전문화된 의사가 많은 양의 의료 영상을 분석하여 진단(레이블)을 작성
  - 비용과 노력...

## 2) AI의 한계 받아드리기 - data

---

- 2. 학습데이터 품질

- (작업자간 통일) 만약 의사마다 진단 기준이 다르다면?
  - 학습 데이터의 일관성이 떨어지고 인공지능 성능 하락
- (작업자의 바이오리듬) 동일 작업자도 매번 동일한 기준을 적용하기는 어렵다.
- (가장 중요) 데이터 수집/파싱/레이블링/검수 전담 부서 필요!!

## 2) AI의 한계 받아드리기 - data

---

- 3. 산업 현장 데이터의 변화

- 데이터의 변화 원인은 다양
- (데이터 자체의 변화) 기존에 없던 새로운 병이 생긴다면?
- (데이터 생성 환경의 변화) 의료 영상 촬영 장비의 변경->의료 영상 이미지 변화
- 데이터의 변화를 감지하고 데이터 추가 구성, 환경 조정 등의 조치가 가능한가?

## 2) AI의 한계 받아드리기 – Possibility??

---

- 애초에 AI가 할 수 있는 일인가??
  - AI가 바둑도 잘하고 글도 써준다는데?
  - 그럼 내가 하는 복잡한 일도 할 수 있는 거 아냐?
    - ~~하고
    - ~~해서
    - ~~를 고려한 후
    - 비교분석을 통해
    - 경험에 의한 판단을 내려서
    - ~~할 때
    - ~~하는 일

## 2) AI의 한계 받아드리기 – Possibility??

- 애초에 AI가 할 수 있는 일인가??

- AI가 바둑도 잘하고 글도 써준다는데?
- 그럼 내가 하는 복잡한 일도 할 수 있는 거 아냐?

- ~~하고

- ~~해서

- ~~를 고지한 후

- 비교분석을 통해

- 경험에 의한 판단을 내려서

- ~~할 때

- ~~하는 일



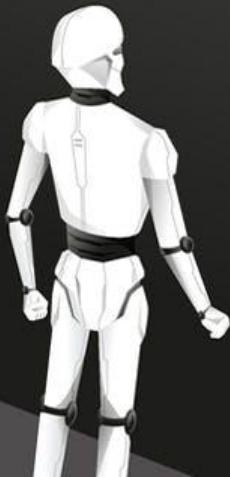
- 제한된 상황

- 학습데이터 분포를 벗어나지 않는 데이터에 한해

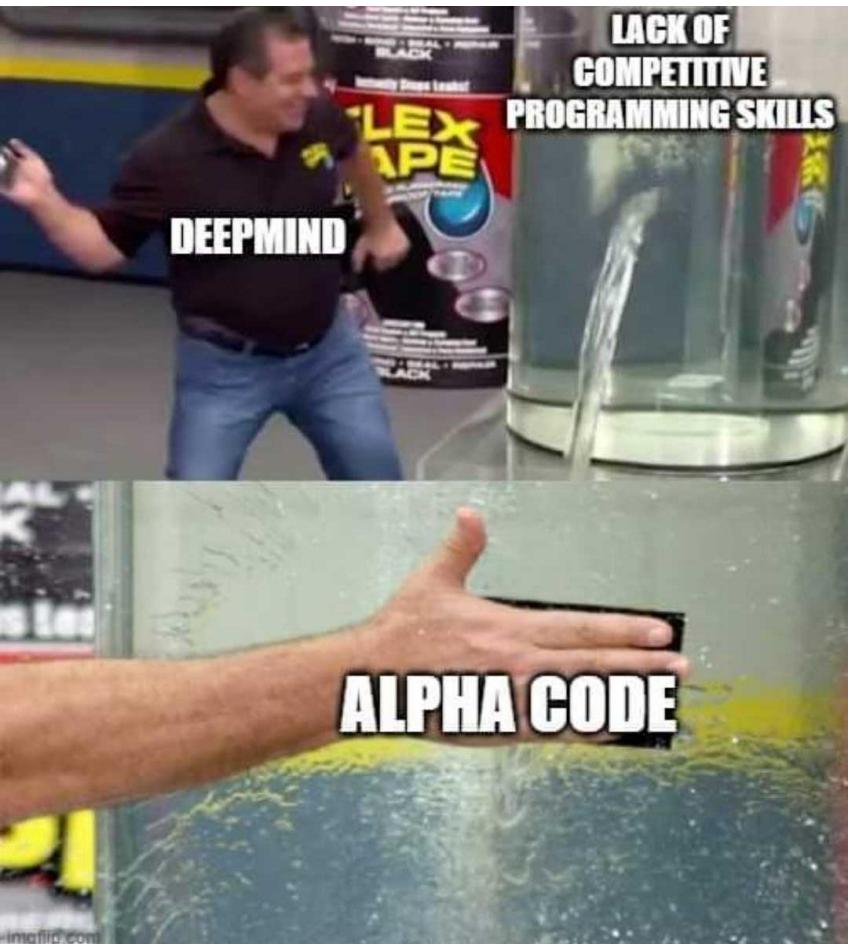
- 지도학습 (분류)

경우에만 **가능**

## 2) AI의 한계 받아드리기 – Possibility??



## 2) AI의 한계 받아드리기 – Possibility??



■ ~~을 때

■ ~~하는 일

### Problem Description

```
# RATING: 1200
# TAGS: sortings
# LANGUAGE IS python3
# CORRECT SOLUTION
# You are given a string s, consisting of n letters, each letter is either 'a' or 'b'. The letters in the string are numbered from 1 to n.
# 
# s[1: r] is a continuous substring of letters from index 1 to r of the string
# inclusive.
# 
# A string is called balanced if the number of letters 'a' in it is equal to the number of letters 'b'. For example, strings "bab" and "aabbab" are balanced
# and strings "aab" and "b" are not.
# 
# Find any non-empty balanced substring s[1: r] of string s. Print its l and r
# (1 ≤ l ≤ r ≤ n). If there is no such substring, then print -1 -1.
# 
# Input
# 
# The first line contains a single integer t (1 ≤ t ≤ 1000) – the number of testcases.
# 
# Then the descriptions of t testcases follow.
# 
# The first line of the testcase contains a single integer n (1 ≤ n ≤ 50) – the length of the string.
# 
# The second line of the testcase contains a string s, consisting of n letters, each letter is either 'a' or 'b'.
# 
# Output
# 
# For each testcase print two integers. If there exists a non-empty balanced substring s[1: r], then print l r (1 ≤ l ≤ r ≤ n). Otherwise, print -1 -1.
```

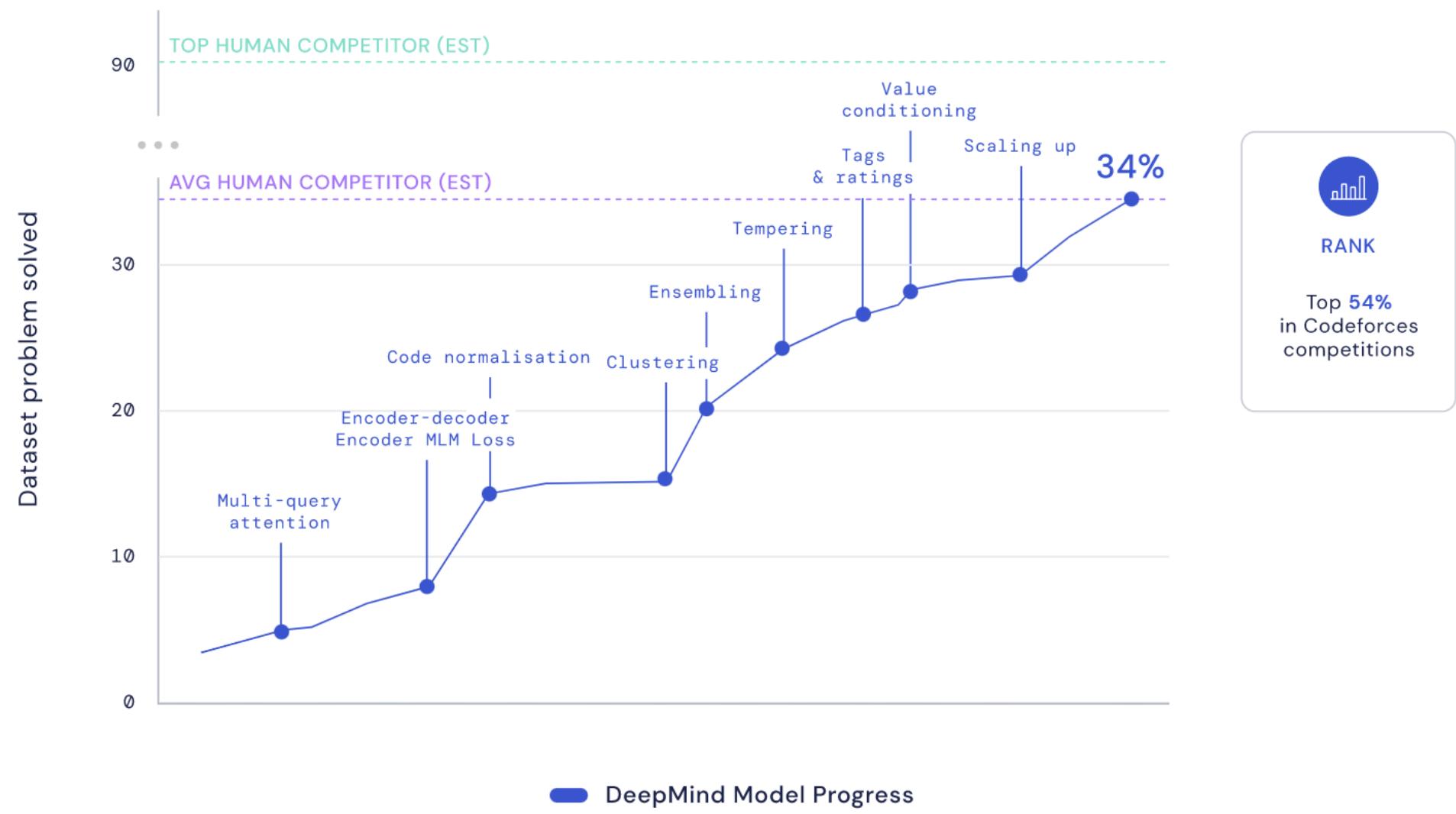
### 할 수

경우에만 가능

### Solution

```
t = int(input())
for _ in range(t):
    n = int(input())
    s = input()
    a = [0] * n
    b = [0] * n
    for i in range(n):
        if s[i] == 'a':
            a[i] = 1
        else:
            b[i] = 1
    flag = 0
    for i in range(1, n+1):
        for j in range(i, n+1):
            if a[i]-a[i-1]==b[j]-b[i-1]:
                print(i, j)
                flag = 1
                break
        if flag==1:
            break
    if flag==0:
        print(-1, -1)
```

## 2) AI의 한계 받아드리기 – Possibility??



## 2) AI의 한계 받아드리기 – Possibility??

### TransCoder

#### Python 함수

```
def sum_elements(values):  
    return sum(values)
```



#### C++ 함수

```
template <typename T> T  
sum_elements(const std::vector<T> &values){  
    return std::accumulate(values.begin(), values.end(),  
                           0);  
}
```

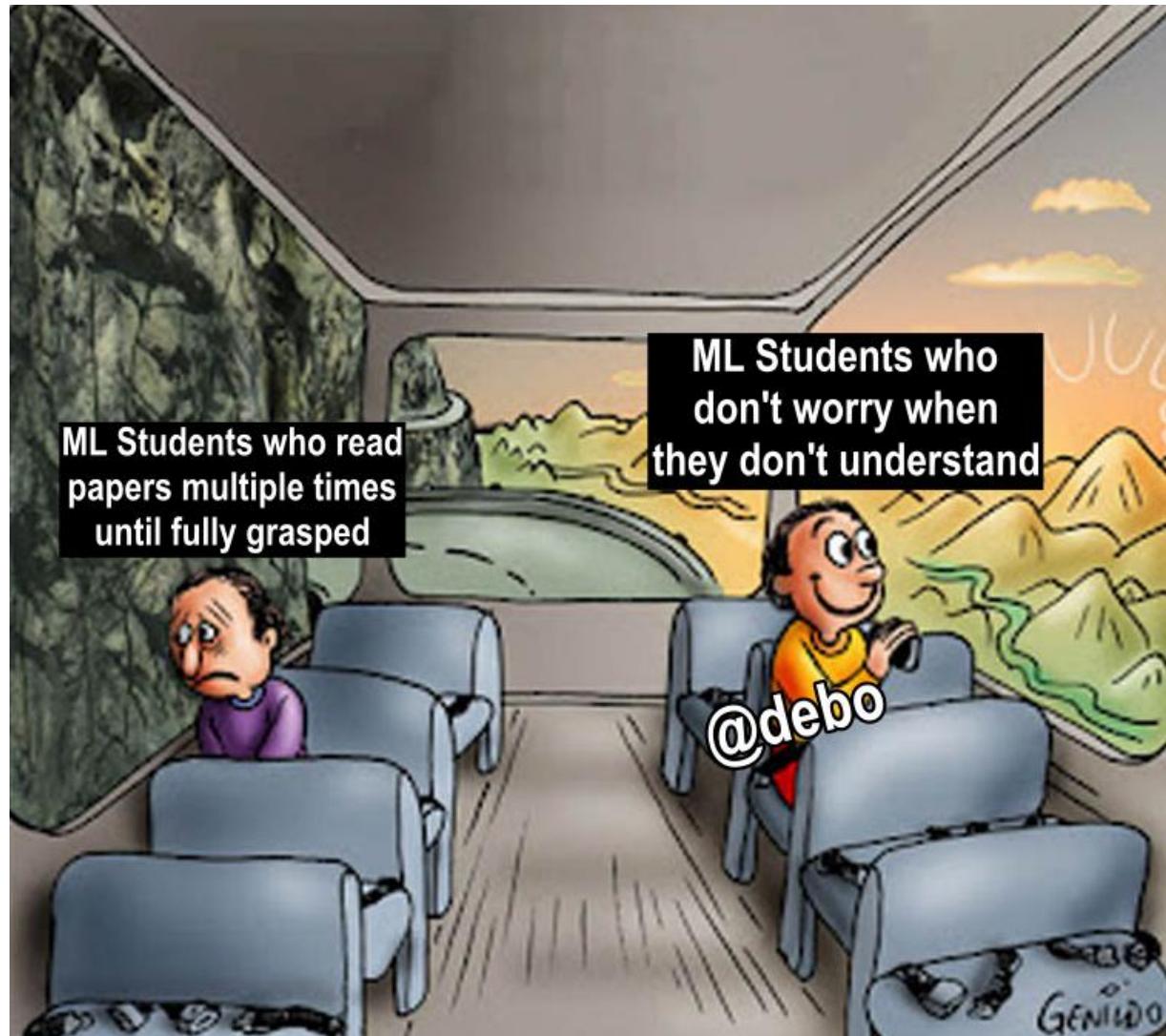
#### TransCoder

```
def no_letters(s):  
    return s.lower() == s.upper()
```

- C++ => Java : 74.8%
- C++ => Python : 67.2%
- Java => C++ : 91.6%
- Python => Java : 56.1%
- Python => C++ : 57.8%
- Java => Python : 68.7%

- 제한된 상황
- 학습데이터 분포를 벗어나지 않는 데이터에 한해
- 지도학습 (분류)  
경우에만 가능

## 2) AI의 한계 받아드리기 – Possibility??

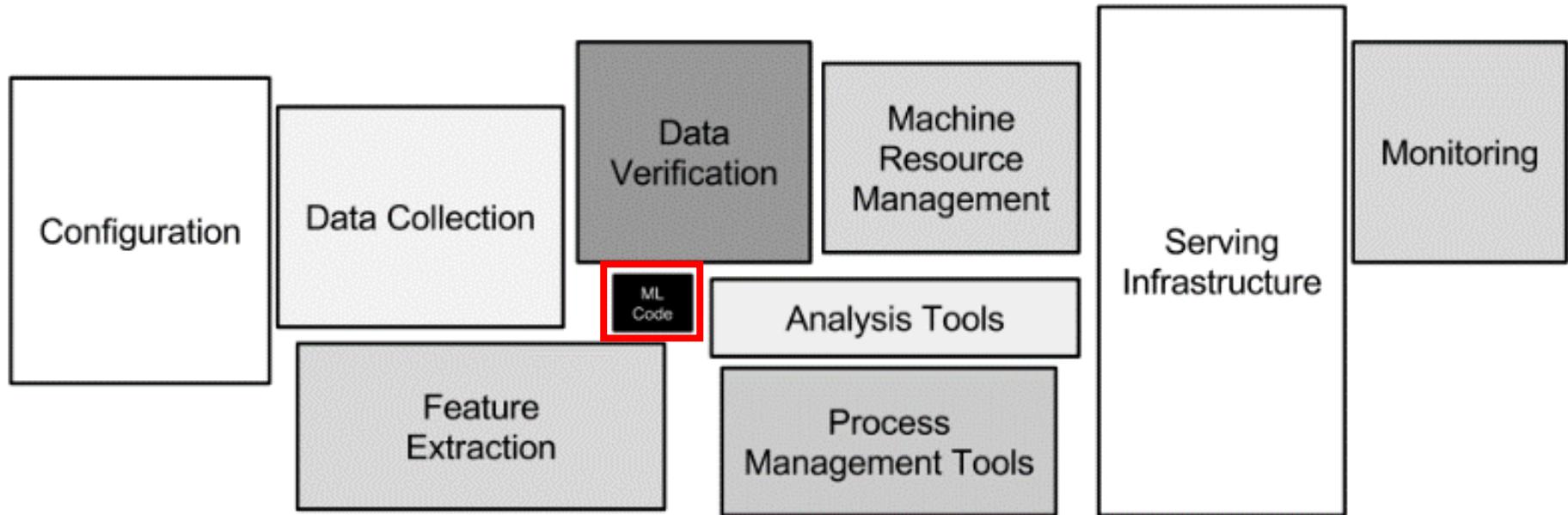


- 제한된 상황
- 학습데이터 분포를 벗어나지 않는 데이터에 한해
- 지도학습 (분류)

### 3) MLOps

- AI시스템에서 AI개발은 아주 일부일 뿐.

- 주변 인프라 업무가 훨씬 크고 복잡하다



### 3) MLOps

클라우드 서버 기반의 AI



요청  
분석



명령  
수행



온 디바이스 AI



명령  
분석/수행



### 3) MLOps

- AI시스템에서 AI개발은 아주 일부일 뿐.

- 주변 인프라 업무가 훨씬 크고 복잡하다

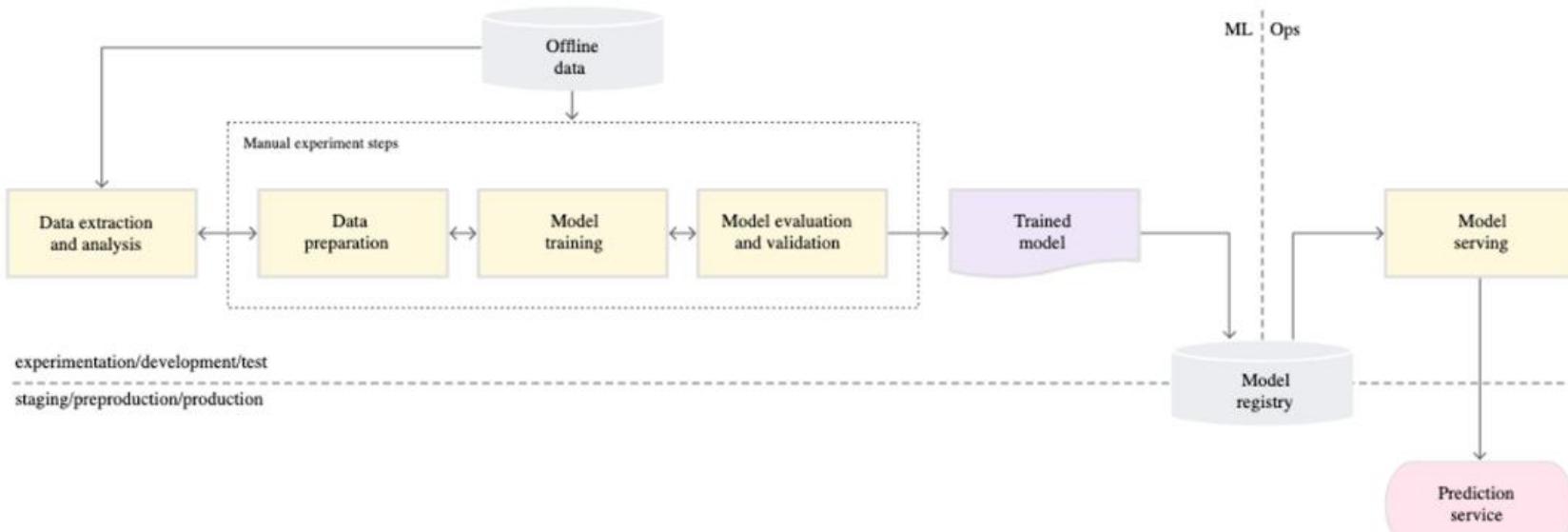


 Kubeflow	 Grafana	 Prometheus	 docker	 kubernetes	 Python
Kubeflow:v1.40	Grafana:v8.1.2	Prometheus:v2.28.1	Docker:20.10.7	Kubernetes:v1.21.2	Python:v3.8
 minikube	 HELM	 DVC	 mlflow	 Flask	 Katib
minikube:v1.22.0	HELM:v3.x	DVC:v2.6.4	mlflow:v1.17.0	Flask:v2.0.1	Katib:v0.120
 Jenkins	 Seldon	 FEAST	 FCML	 Github Actions	 ML OPS
Jenkins:v2.30.3	seldon-core:v1.11.0	FEAST:v0.12	FCML:v0.6.3	Github Actions:v1.1.0	ML OPS:v1.0

### 3) MLOps

#### ■ AI시스템에서 AI개발은 아주 일부일 뿐.

- 주변 인프라 업무가 훨씬 크고 복잡하다
- ML+Ops의 협업 필요
- **Ops 전담팀 필수!!!**

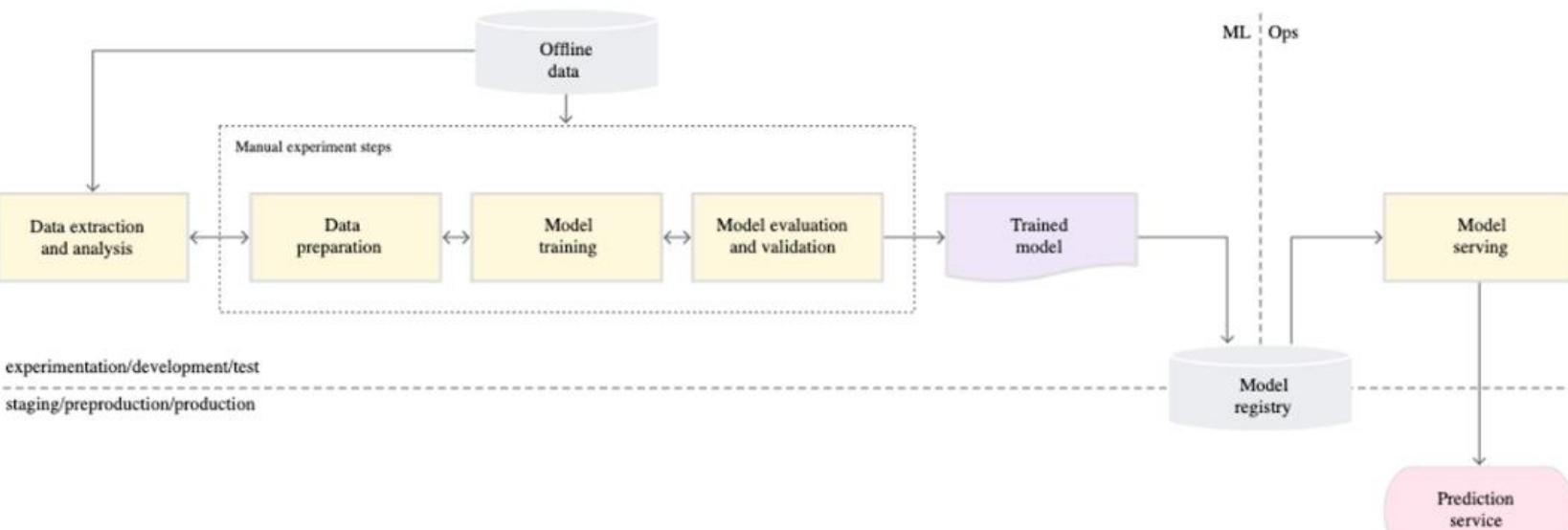




### 3) MLOps

#### ■ AI시스템에서 AI개발은 아주 일부일 뿐.

- 주변 인프라 업무가 훨씬 크고 복잡하다
- ML+Ops의 협업 필요
- **Ops 전담팀 필수!!!**



새 모델입니다

확인해 보겠습니다

이거 안되는데요?

어.. 저는 되는데요?

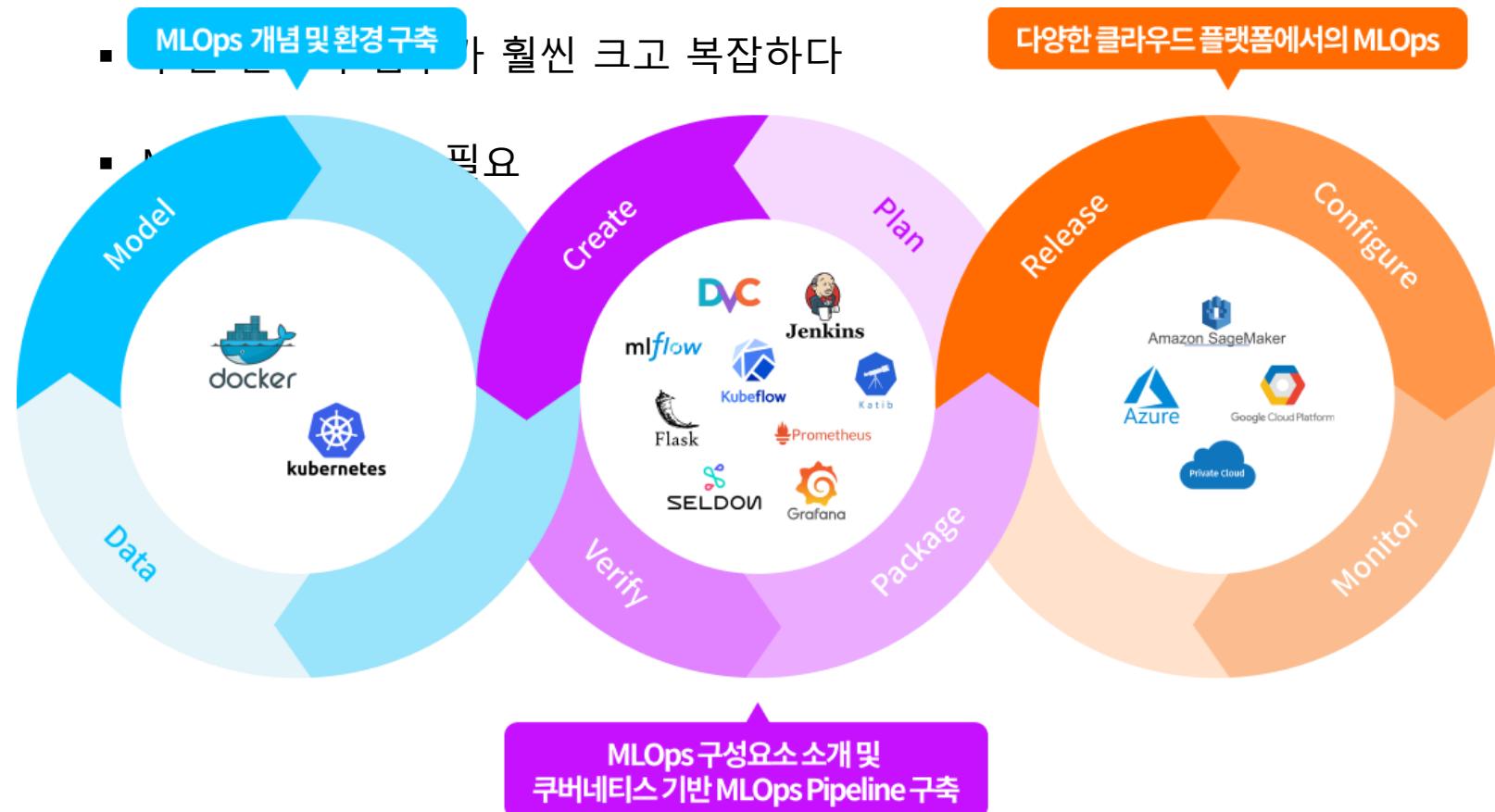
파이썬 뭐 쓰세요?

아! 제가 쓰는게 파이썬 버전  
호환이 안 되어서 3.8로 올렸어요!

아.. requirements.txt 좀  
보내주세요..

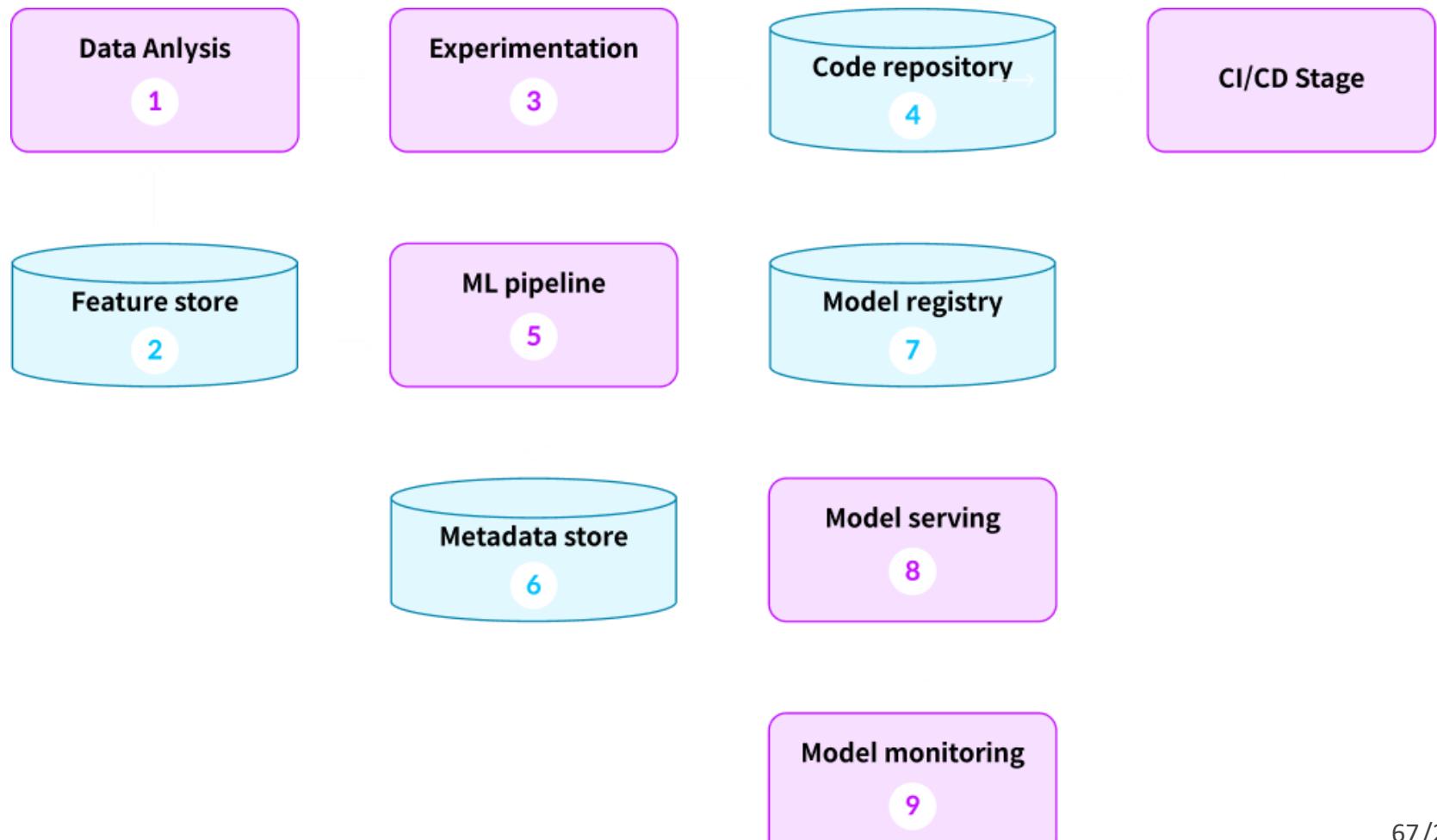
### 3) MLOps

- AI시스템에서 AI개발은 아주 일부일 뿐.

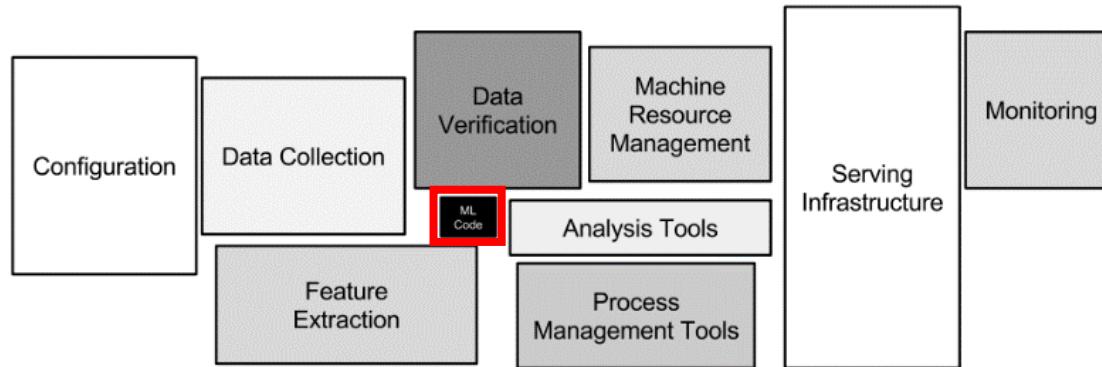


# 3) MLOps

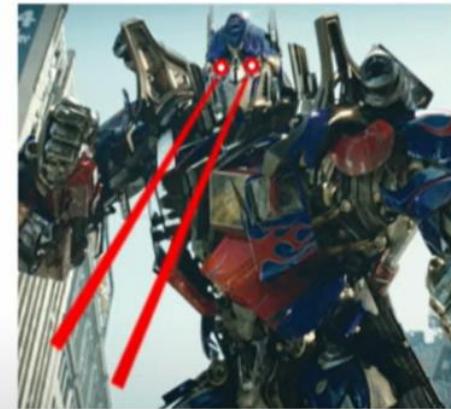
## 머신러닝(MLOps) 파이프라인



### 3) MLOps



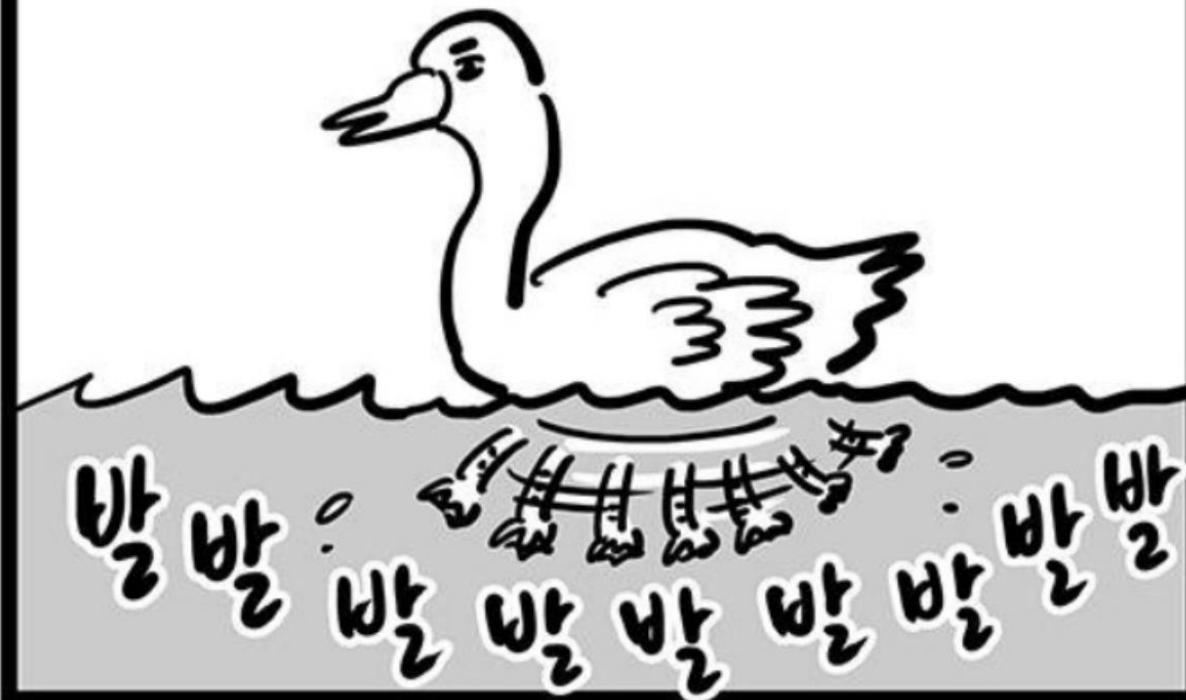
“컨테이너가 너무 많아진다..!”



“서버야 너는 지금 뭐해 자니? ”

### 3) MLOps

마치 백조가 수면 위의 모습은  
평온해 보이지만 수면 아래는  
치열하게 움직이는 것처럼!



## 4) AI CTO

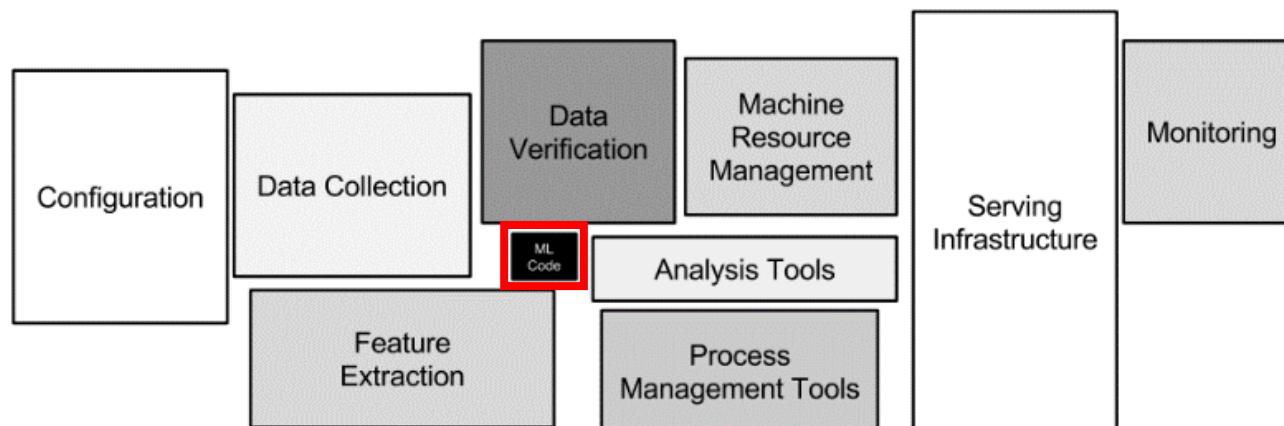
- AI시스템을 위한 군단이 필요

- AI

- MLOps

- 빅데이터 시스템 구축

- DATA 담당



## 4) AI CTO

- AI시스템을 위한 군단이 필요

- Frontend
- Backend
- App
- DevOps



## 4) AI CTO

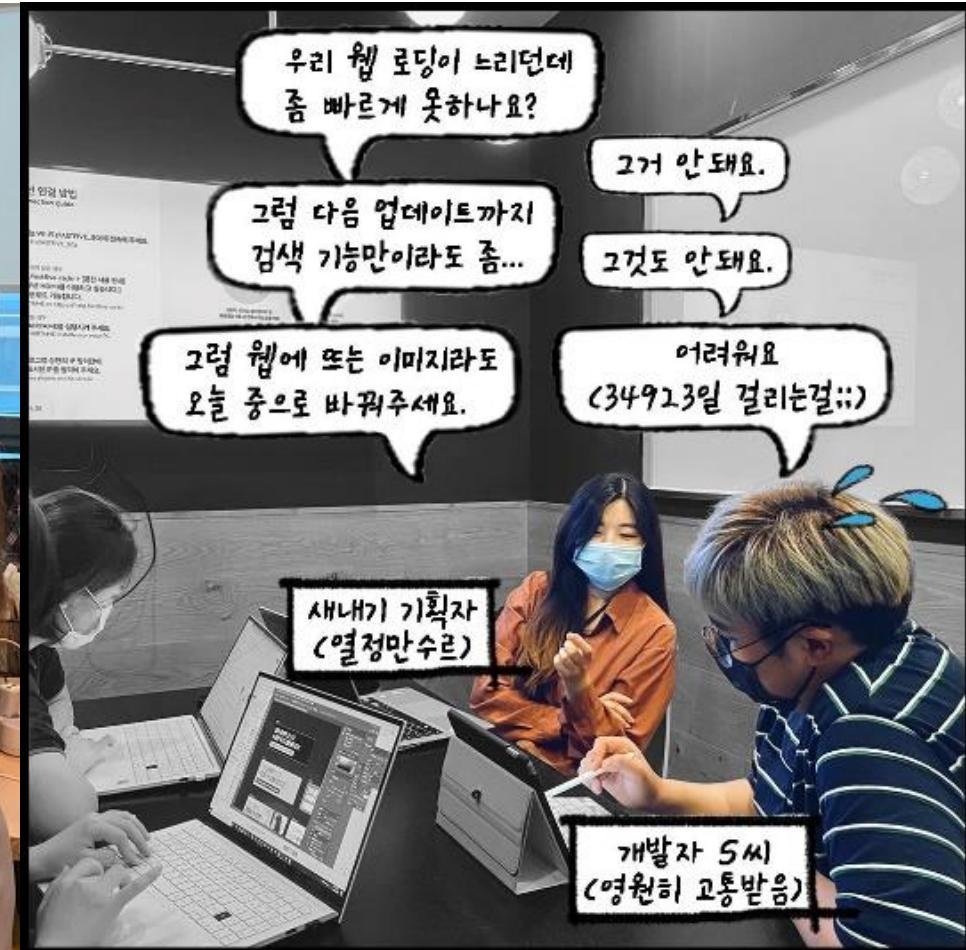
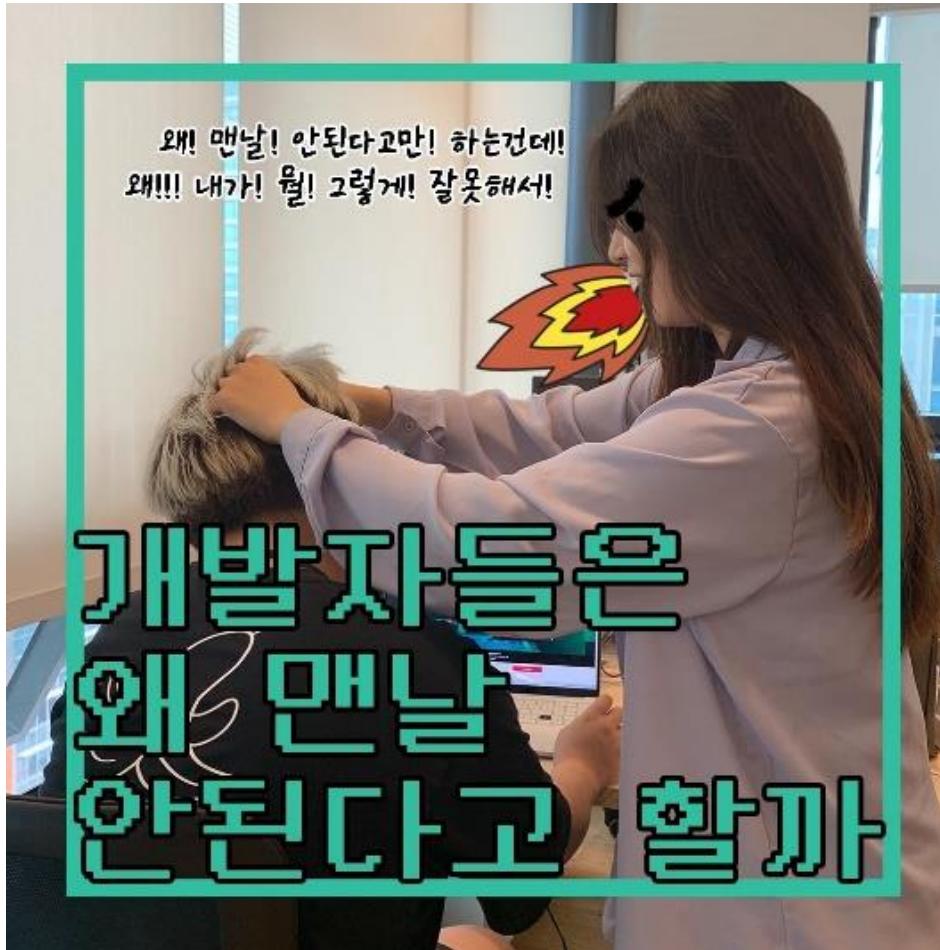
- AI시스템을 총괄하는 CTO는

- AI
- MLOps
- 빅데이터 시스템 구축
- DATA 담당
- Frontend
- Backend
- App
- DevOps



이 모든 과정을  
단 한 부분도 소홀히 하지 않고  
조율하고 백업할 수 있어야 한다

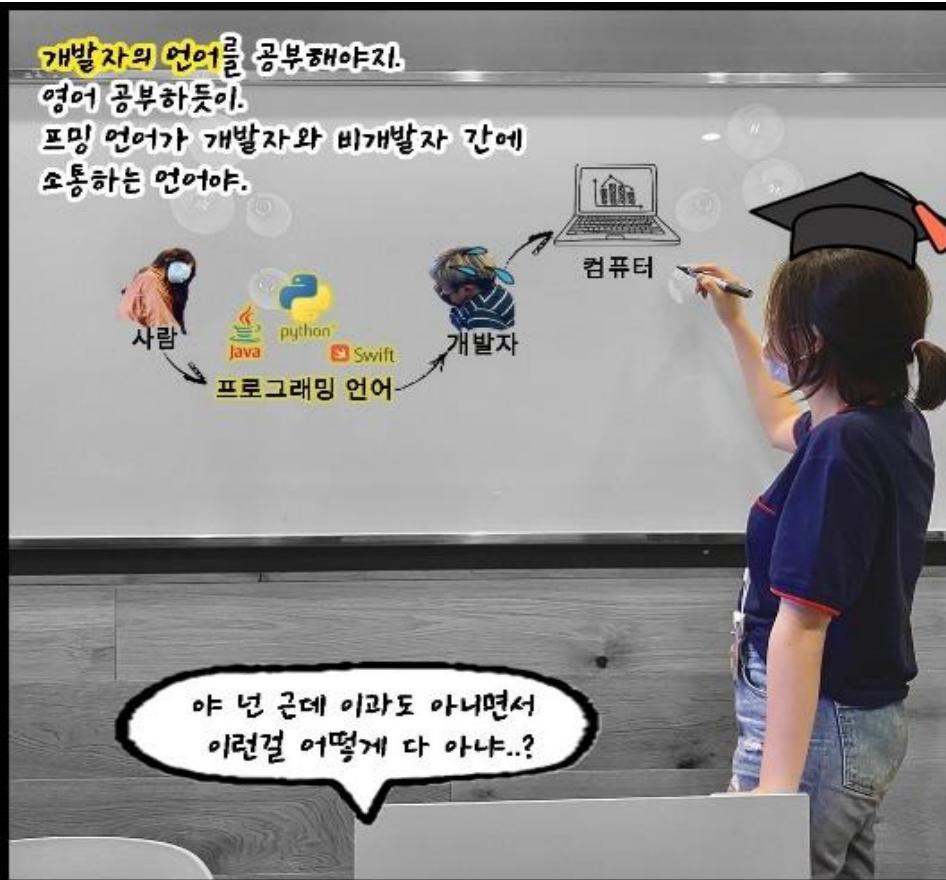
## 4) AI CTO



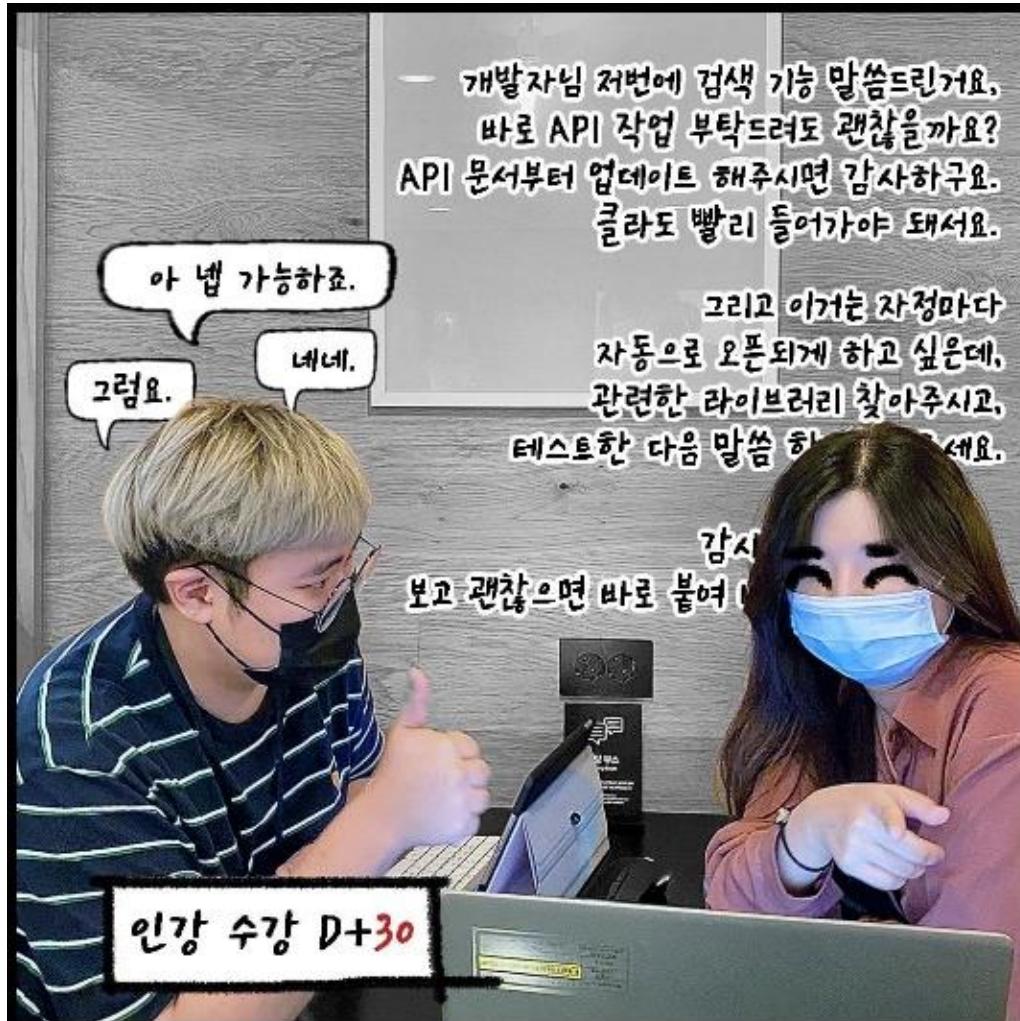
## 4) AI CTO



## 4) AI CTO



## 4) AI CTO



정치만 있던 기획자 K씨가  
도통 할 줄 모르는?



내가 개발자와 협업하는  
기술 / 디자인 / SEO라면  
지금 바로 수강신청하러 가기

## 4) AI CTO

---

- AI시스템을 총괄하는 CTO는
  - AI시스템을 위한 전체적인 이해
  - 회사가 보유한 데이터 확인하여 실행 가능한 AI프로젝트를 물색
  - AI가 적절한지??
  - AI로 할 수 있는 것인지??
  - 적합한 인재 모집
  - 도메인 전문가와 협업
  - 사내 팀구조 빌딩
  - 사업부서 간 협업 및 구조혁신

이 모든 과정을  
단 한 부분도 소홀히 하지 않고  
조율하고 백업할 수 있어야 한다

## 4) AI CTO

- AI시스템을 총괄하는 CTO는
  - 좋은 논문을 엄청 많이 쓴 젊은 교수님?
  - AI를 25년간 연구하신 연륜 있는 교수님?
  - 이곳 저곳 많은 곳에서 CTO를 하신 경력이 화려한 분?
  - 트렌드 책을 써서 유명한 강사?
  - AI 코딩도 해봤고, 기업에서 몇몇 프로젝트를 성공하여 서비스 및 사업화를 성공한 경력이 있는 기업출신?
- AI
- MLOps
- 빅데이터 시스템 구축
- DATA 담당
- Frontend
- Backend
- App
- DevOps

신기술을 파악하여 코드 구현능력 (AI에 대한 이해)

AI 프로젝트 성공 경력 (AI프로젝트를 위한 빅데이터 시스템, 데이터 구축, 데이터 정제 이해)

서비스/사업화 경력 (AI를 서비스 하기 위한 MLOps, Fron/BackEnd/APP 등에 대한 이해)

## 5) AI 인재 영입

# 인공지능 시대의 경쟁력은 인재...기업은 AI 인재에 투자하라!

▲ 조행만 기자 ◎ 입력 2022.01.30 17:13 ■ 댓글 0 ♥ 좋아요 0



CFO 57%, AI 및 ML 기술 있는 사람...신규 채용 계획 밝혀  
훌륭한 임무가 훌륭한 인재 끌어모아...훌륭한 연봉 중요해  
ROTC를 'ROTC+'로 수정해 AI 인재 파이프라인 구성 제안



## 5) AI 인재 영입

---

- 현재 전 세계적으로, 기업들은 AI 인재 확보에 혈안
- 국내 기업 50곳을 대상으로 한 AI 인재 관련 설문 조사 결과
- 인공지능 관련 인재가 필요하다?
  - YES! (92%)
  - But, 채용에 적합한 인재가 없다 (68%)
    - 전 세계에서 고난이도 AI 연구개발 인력은(교육&기술) 약 1만 명에 불과하다 by 뉴욕타임스

## 5) AI 인재 영입

---

- AI 인재양성이 필요한가?
  - Yes (4%)
  - 중립 (36%)
  - No (46%)
  - Never (14%)
- AI 인재 양성 시 어려운 점은?
  - 연봉을 맞추기 어렵다 (36%)
  - 적합한 인재가 없다 (68%)
  - 빨리 이직한다 (22%)
  - 재교육이 어렵다 (4%)

## 5) AI 인재 영입

### ▪ 하늘에 별 따기

#### Responsibilities:

언어모델의 적용/개발/최적화

- 인더스트리별 문제를 명확히 정의하고 문제해결을 위한 방법론을 확인하여 적용
- 거대 언어모델(GPT-3 등)을 비즈니스에 적용하고 서비스 환경에 알맞게 튜닝 수행
- 다양한 데이터 소스의 분산/병렬처리, 언어모델의 학습/추론 알고리즘 최적화

#### Qualifications:

[Minimum Qualifications]

- 관련 분야 석사 학위 이상 소지, 3년 이상 실무 경력 보유
- Deep Learning 알고리즘 연구/개발 경험 보유
- Python, C/C++, Java 프로그래밍 언어의 숙련된 활용 능력
- 자연어처리 기술에 대한 이해와 코드 구현 능력(Tensorflow, PyTorch, Keras)

[Preferred Qualifications]

- 관련 분야 박사 학위 소지, 3년 이상 실무 경력자
- DL 기반의 NLP 알고리즘을 직접 서비스에 적용하고 상용화한 경험
- Transformer 구조의 BERT, GPT 모델 적용 경험
- 특정 Industry에 대한 높은 이해
- 최신 AI 연구기술/논문에 대한 이해 및 구현 경험

하늘의 별 따기...



# 5) AI 인재 영입

## ▪ 하늘에 별 따기



### 담당업무

- AI 알고리즘 연구 개발
- AI 모델링 비지니스 적용
- AI 학습 데이터 설계 및 분석

### 자격요건

- Python 언어로 논리 구현 가능하신 분
- Pytorch, Tensorflow 등의 Deep learning 프레임워크를 사용해보신 분
- 인공지능 관련 교육과정 이수 또는 관련 연구소나 대학원 Lab에서 인턴 경험이 있으신 분

### 우대사항

- 관련 분야 전공자
- 관련 분야 석사/박사 과정 졸업자



### 모집부문

### 담당업무

### 자격요건 및 우대사항

#### AI 연구원

[포지션]  
- LAB실, 대리~과장급

[담당업무]  
- 플랫폼 알고리즘 연구 및 개발  
- 플랫폼 서비스 전반적인 운영

[자격조건]  
- 경력 3 ~ 10년 선호  
- Java, Spring 숙련자  
- "인공지능 및 추천" 관련 업무 경험자

[우대사항]  
- 능동적인 업무 태도와 강한 추진력  
- 라이브러리 또는 팀원에 의존하기보다 주도적인 업무 역량 보유

[기타사항]  
- 정규직, 주 5일근무  
- 근무지 : 서울 구로구

## 5) AI 인재 영입

### ▪ Oh yeah, recruiting. Maybe they are looking for BERT's authors?

Responsibilities/Experience:

- Masters / PhD in a quantitative and technical discipline
- Must have **4+ years** of applied NLP experience using **RoBERTa** or **BERT**
- Proficiency with Python
- Deep knowledge of NLP concept extraction strategies / tools:  
RoBERTa, BERT

<https://arxiv.org> > cs ▾

#### RoBERTa: A Robustly Optimized BERT Pretraining Approach

by Y Liu · 2019 · Cited by 1457 — Computer Science > Computation and Language.

arXiv:1907.11692 (cs). [Submitted on 26 Jul 2019]. Title: **RoBERTa**: A Robustly Optimized BER...

<https://arxiv.org> > cs ▾

#### BERT: Pre-training of Deep Bidirectional Transformers for ...

by J Devlin · 2018 · Cited by 21997 — We introduce a new language representation model

called **BERT**, which stands for **Bidirectional Encoder Representations from Transformers**...

## 5) AI 인재 영입

- 하늘에 별 따기

- AI
- MLOps
- 빅데이터 시스템 구축
- DATA 담당
- Frontend
- Backend
- App
- DevOps



## 5) AI 인재 영입

- 어차피 내가 원하는 풀스텍 개발자는 세상에 존재하지 않는다



## 5) AI 인재 영입

- 자연어처리 전공 경력사원을 채용했다
- 챗봇도, 번역기도, 요약모델도, 검색시스템도 척척 만들 수 있을 줄 알았다
- 근데 GPT(언어생성모델)만 만들어봤다고 한다??



## 5) AI 인재 영입

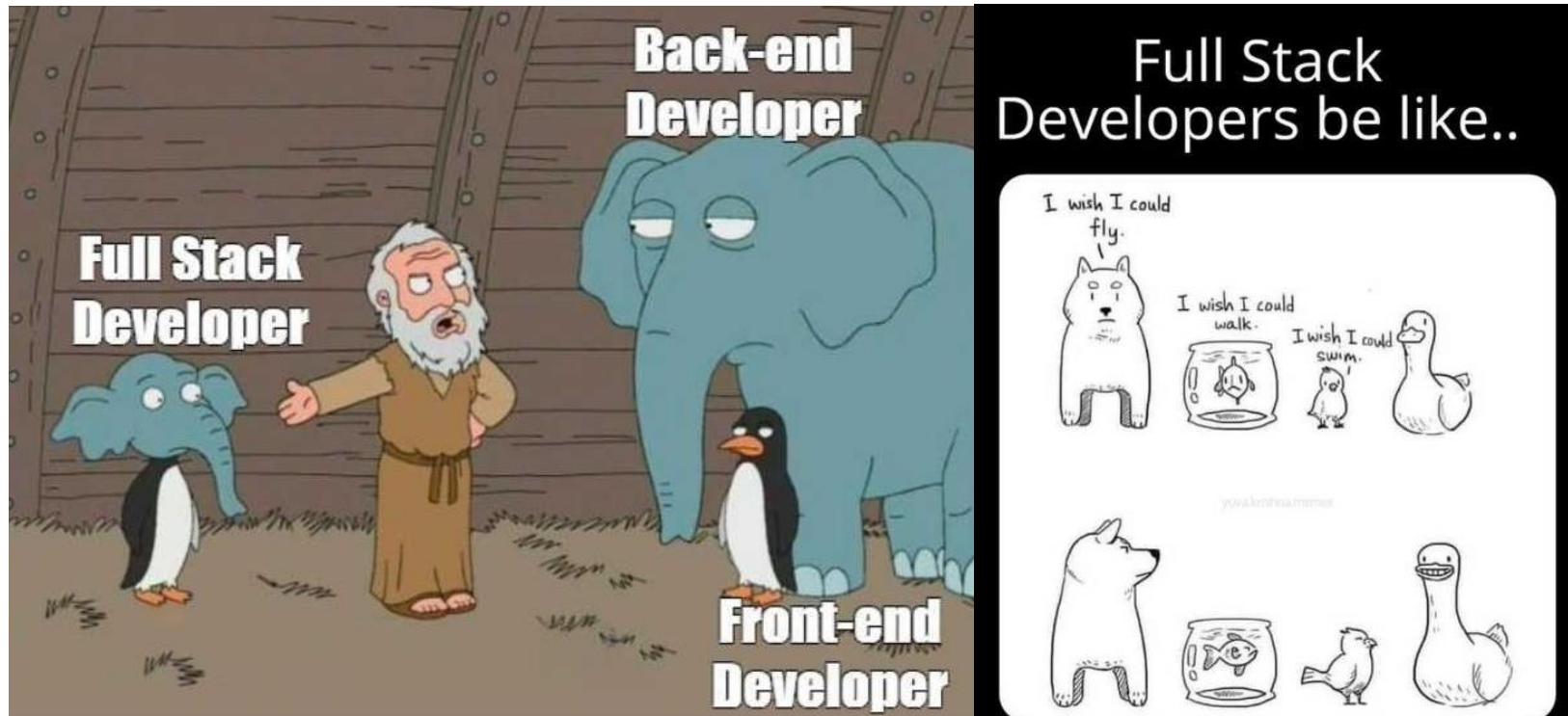
- 자연어처리 전공 경력사원을 채용했다
- 챗봇도, 번역기도, 요약모델도, 검색시스템도 척척 만들 수 있을 줄 알았다
- 근데 GPT(언어생성모델)만 만들어봤다고 한다??

- 그럼 GPT라도 만들어보라고 했다
- 데이터가 없다고 한다
- 1억짜리 GPU서버로 안된다고 한다
- 혼자선 못한다고 한다
- 뭐지??



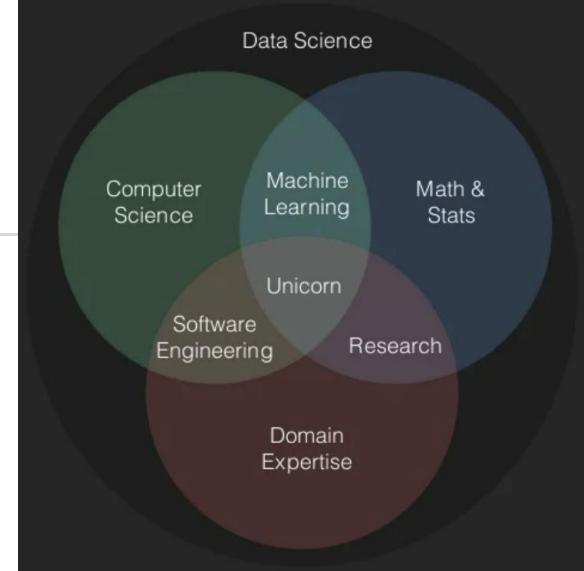
## 5) AI 인재 영입

- 어차피 내가 원하는 풀스택 개발자는 세상에 존재하지 않는다
- 해본게 많지만 경험이 적은 풀스택들은 실제로 깊이가 모자란 경우가 많다



## 5) AI 인재 영입

- 정말 아주 가끔 유니콘이 있다!
- 이들을 발견하면 몇억을 줘서라도 꼭 잡아야 한다
- 뛰어난 개발자는 혼자 5인~10인분을 하기도 한다



## 5) AI 인재 영입

- 필수 고려

- 재택근무 여부
- 자율출퇴근
- 자기발전 기회(논문)

- 연봉

- 성과에 따른 확실한 인센티브

- 경계

- '같이/함께 성장하는'

- '가족같은 분위기'

입사전



입사후



## 5) AI 인재 영입

# Deep Learning



What society thinks I do



What my friends think I do



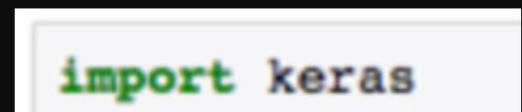
What other computer  
scientists think I do



What mathematicians think I do

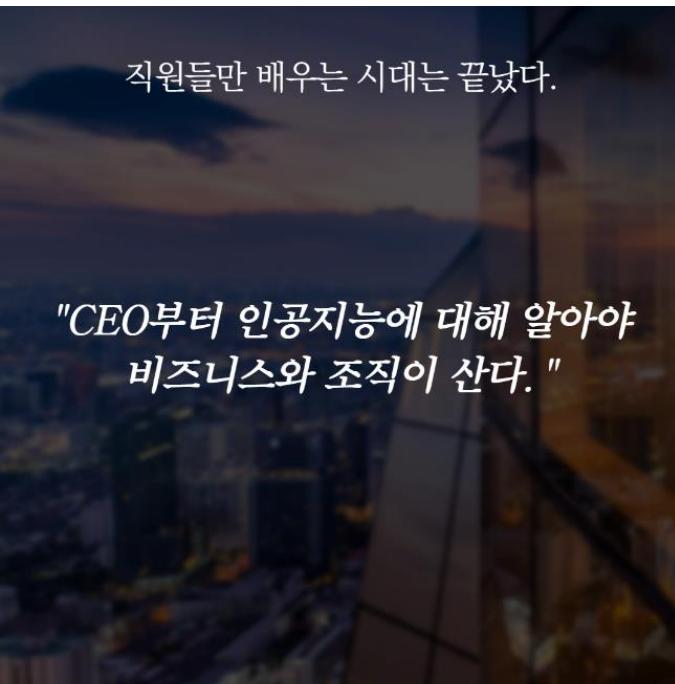


What I think I do



What I actually do

## 6) 사내 팀구조



## AI로 일하는 기술 인공지능은 어떻게 일이 되는가



★★★★★ 0.0 | 네이버리뷰 20건

저자 장동인 | 한빛미디어 | 2022.01.03

페이지 392 | ISBN ? 9791162244913 | 판형 규격외 변형

도서 16,200 원 18,000 원 -10%

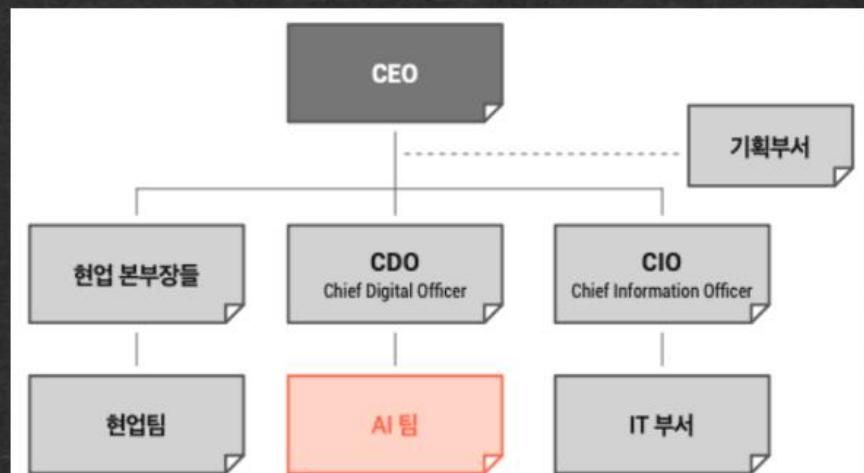
판매 12,960 원 14,400 원 -10%

구매혜택 살펴보기 >



# 6) 사내 팀구조

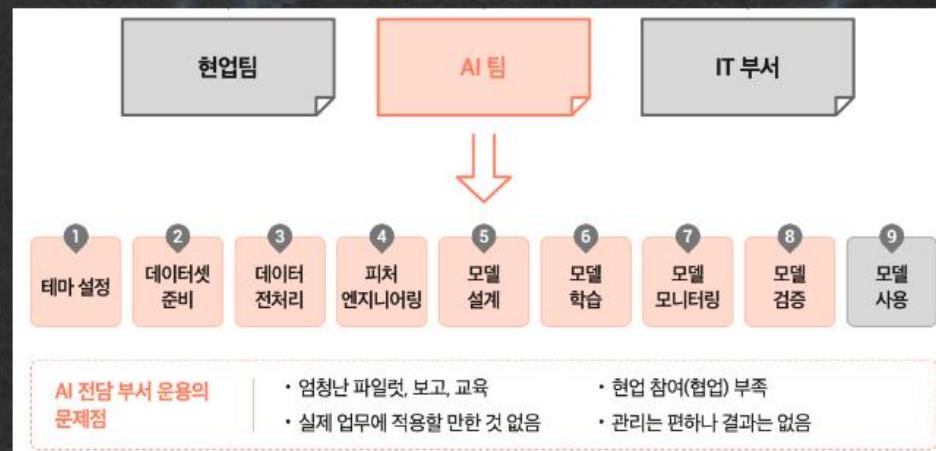
\*NEW 조직도\*



신중한 고민 끝에

'외부 전문가로 구성된 AI팀을 신설'했습니다.

그런데

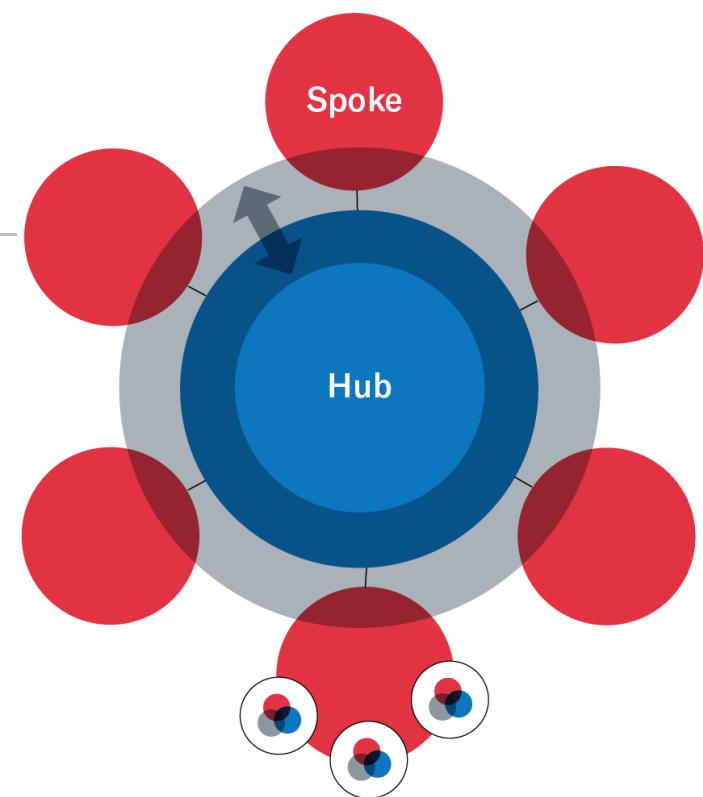


"AI팀과 다른 조직간의 업무 협조 부재"

의 이유로 성과를 거두지 못하고  
팀이 해체되었습니다.

## 6) 사내 팀구조

- 왜 현업 부서와 업무협조가 안되지?
  - 인원부족/의심/고유업무/공
  - '인공지능 부서'를 달가워하지 않는다
  - '알아서 협조해'? 필패

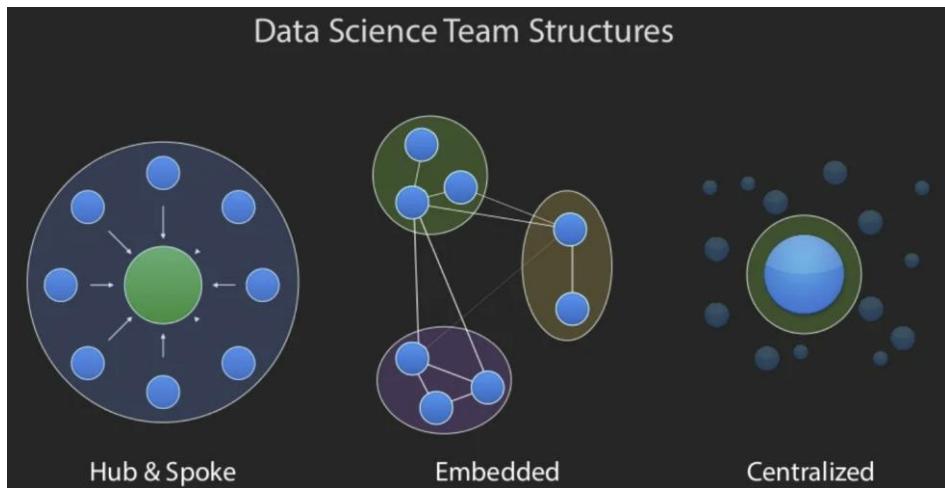


Spoke

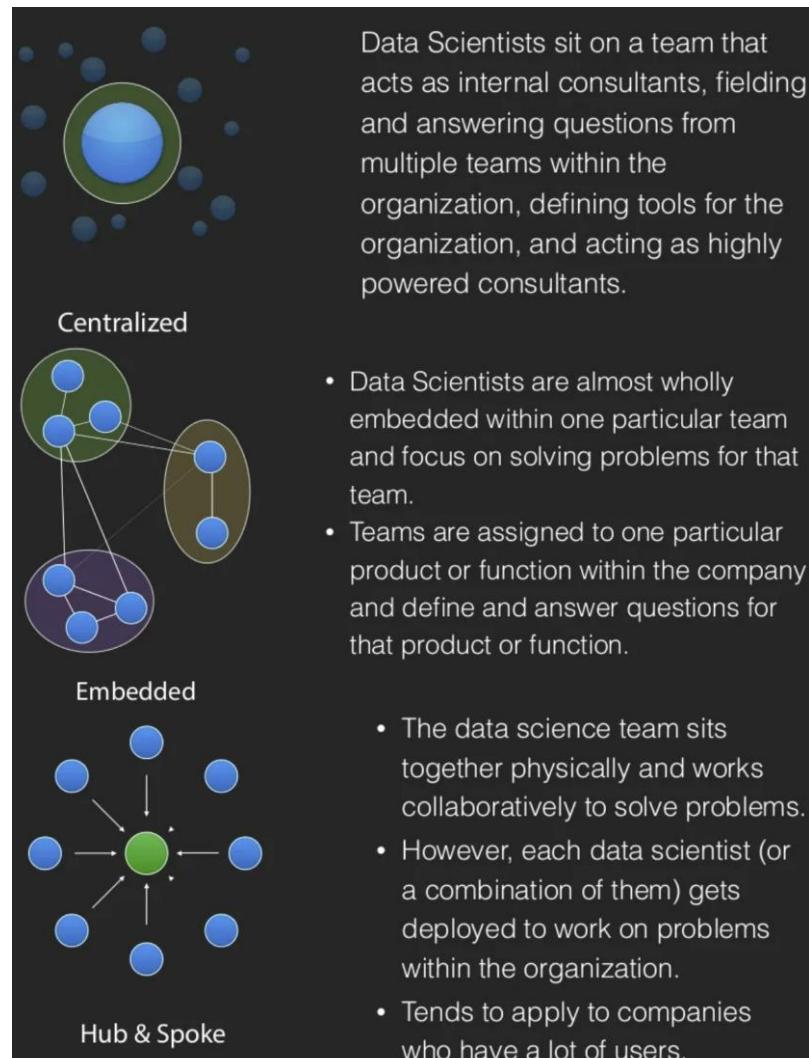
## 6) 사내 팀구조



# 6) 사내 팀구조



- 데이터 기반 결정이 직감이나 전통보다 우월하다는 믿음을 쌓는 것
- 관리자나 임원진은 일반 통념을 고수하고 데이터에 대한 신뢰성이 부족
- 의사결정 권한을 애널리틱스 프로세스에 양도하기를 거부

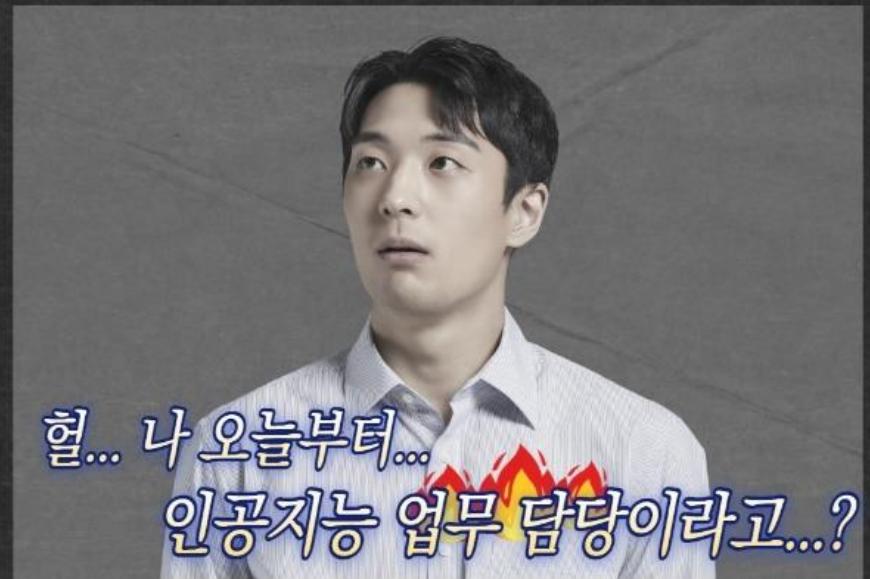


## 6) 사내 팀 구조



이번에는 반드시 성공하겠다는 다짐으로  
개발 좀 한다는 내부 직원들로  
AI팀을 다시 재정비하였습니다.

하지만



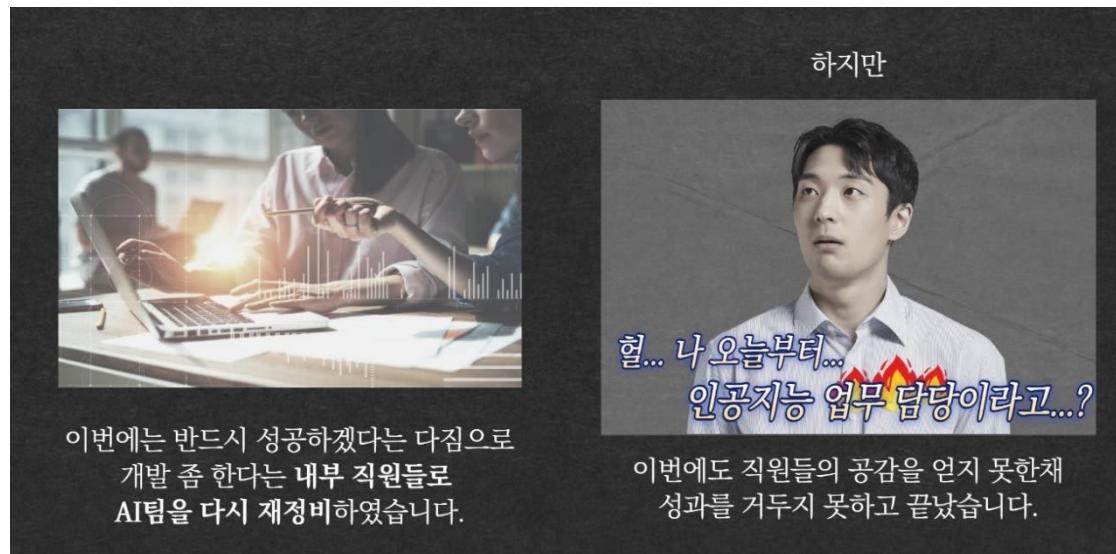
이번에도 직원들의 공감을 얻지 못한채  
성과를 거두지 못하고 끝났습니다.

# 6) 사내 팀 구조

- 내부직원 교육이 실패하는 이유

- 수년간 공부/연구/개발 하던 주제를 두고
- 갑자기 AI연구를 하는 도전은 쉽지 않음
- 자기 지식에 갇혀 빠져나오기 힘듦

- 도메인 전문가와 AI전문가를 그룹화하고 융합/협업을 철저히 관리



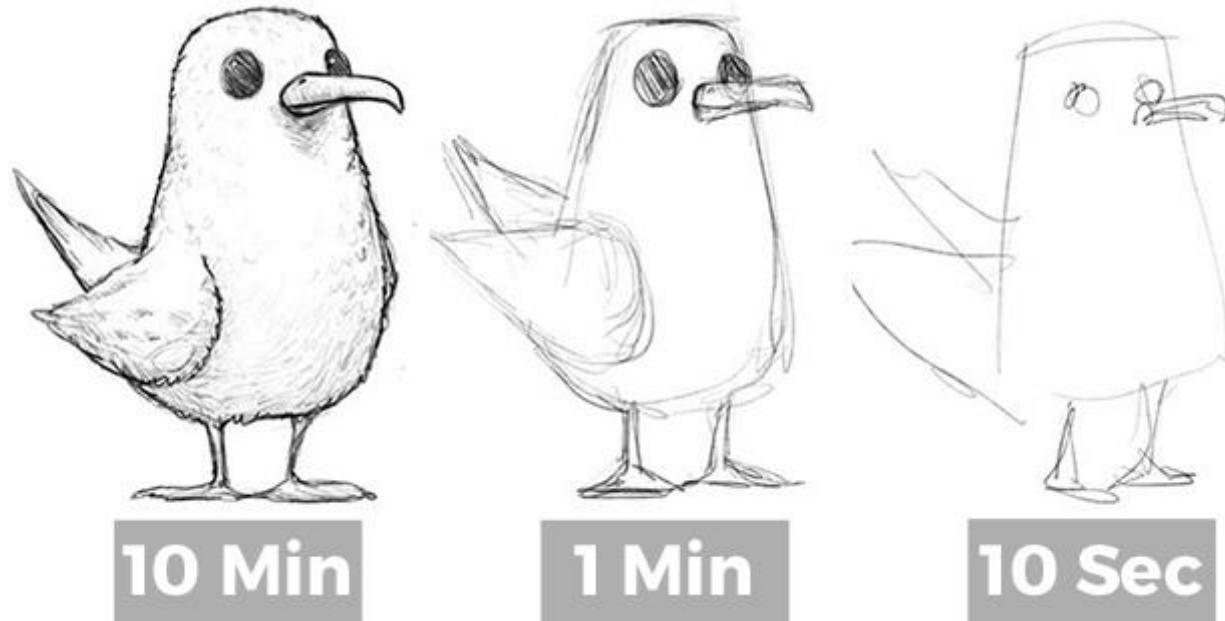
## 7) 실패한다면? 100% C-level 때문이다



주연

## 7) 실패한다면? 100% C-level 때문이다

- 아주 흔한 실패 과정 1) 인력/기간 할당
  - CTO> “XX님 자연어처리 전공하셨죠? 챗봇 하나 만들어주세요”
  - AI> 네?? 챗봇이요? 저 챗봇은 모르는데...
  - CTO > “**6개월** 줄게요, **한달** 후 **대충 윤곽** 만들어서 보고하세요”



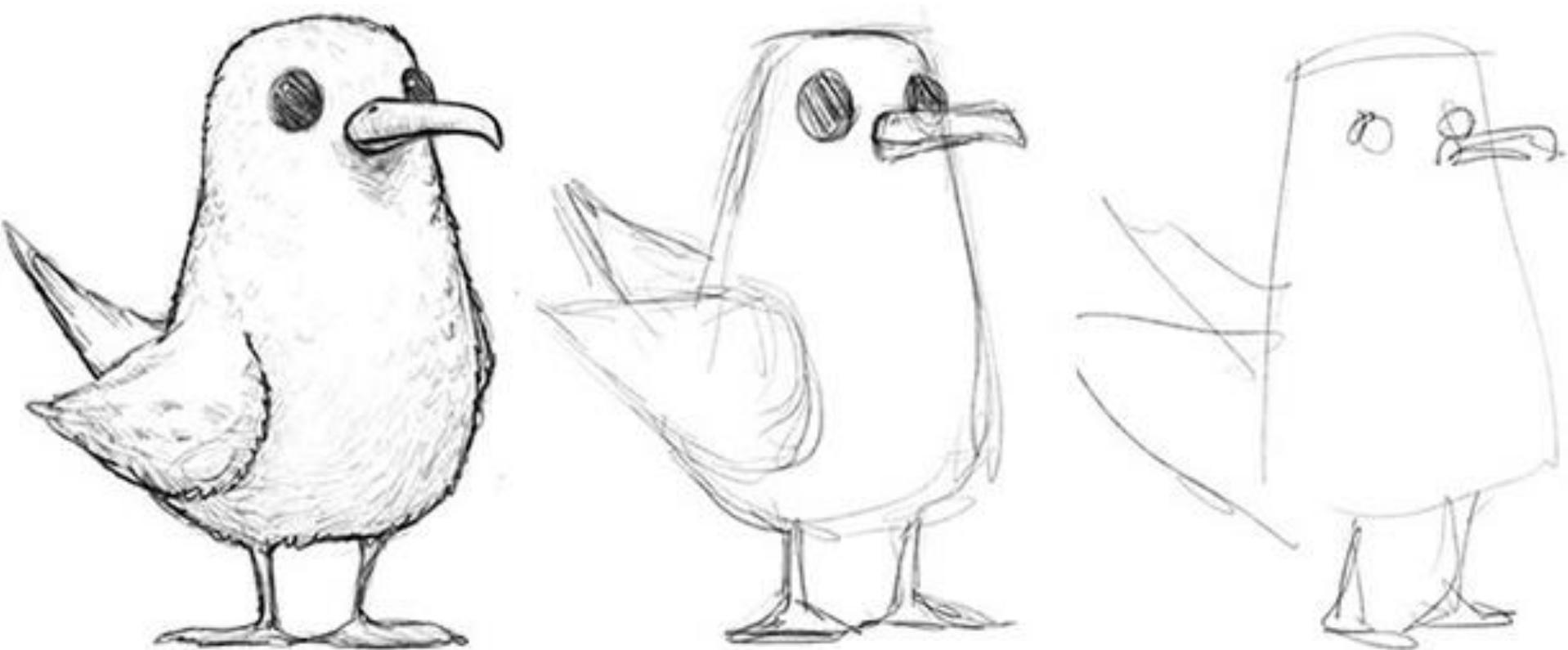
# 7) 실패한다면? 100% C-level 때문이다

---

- 인력/기간 할당
- 그래서 몇명이 얼마짜리 GPU로 몇달이 소요되지??
  - > 컨설팅 필요 (최소 3분의 전문가에게 따로 상담)
    - 최종 목적은 '서비스'
    - 연구 위주인 '교수'님들보단 실제 서비스 중인 전문가에게
- 예시) 챗봇/번역/GPT
  - 데이터팀 (구축/정제/검수)
  - AI모델팀 (GPT/S2S 등 모델개발)
  - 인프라팀 (실시간 챗봇응답서비스 담당)

## 7) 실패한다면? 100% C-level 때문이다

- 인력/기간 할당



10 Min

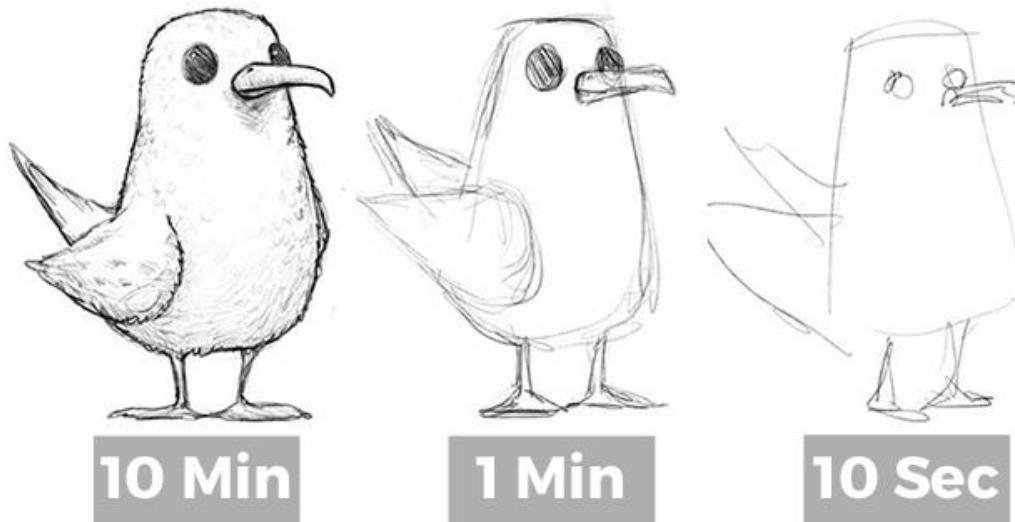
1 Min

10 Sec

# 7) 실패한다면? 100% C-level 때문이다

- GPU서버

- AI> GPT로 챗봇만드려는데 GPT학습하기에 서버가 부족합니다..
- CTO > GPU? 작년에 좋은거 500만원짜리 8대 사지않았어요?
- AI> 그거론 안됩니다...
- CTO > 매년 사는데도 GPU가 부족하다고요??



# 7) 실패한다면? 100% C-level 때문이다

- 아주 흔한 실패 과정

- “서버에 데이터 많자나요?”
- “언제까지 데이터 없다고 할거에요??”
- “데이터 없어도 학습할 수 있다는데?”

데이터 노동자 5인의 이야기

신OO  
(41 · 전업주부)  
자율주행 AI 학습용 데이터 라벨링  
"애들이 학교갈 땐 그 시간 활용해"

김OO  
(38 · 휴직)  
딥페이크 방지 AI 학습용 영상촬영 모델  
"코로나 시대에 맞는 일 같아!"

이준수  
(25 · 취업준비생)  
안면인식 AI 학습용 데이터 라벨링  
"앉아서 머리로 하는 단순 반복 노동"

신단비  
(20 · 캐나다 유학생)  
뇌신경세포 3D 이미지 데이터  
추적 라벨링  
"한국에서 온라인 강의 들으면서  
아르바이트"

박OO  
(31 · 프리랜서  
번역가)  
번역 AI 학습용 데이터 검수  
"업으로 하기에는 안정성이 떨어  
진다 생각"

## 7) 실패한다면? 100% C-level 때문이다

- 아주 흔한 필패 과정
  - “바둑도 두고 게임도 한다는데 우리AI팀은 대체 뭐 하는 거야?”
  - “Tensorflow 인가 뭔가 그거 가져다 그냥 돌리면 되는 거 아냐?”
  - “구글에서 다 해놓은 거 가져다 쓰면 된다면서”
  - “요즘 No코딩으로 고등학생들도 AI 만든다는데??”
  - “내가 원하는 거 물어보면 대답해주는 챗봇 하나 만들어줘요”
  - “내가 보고 싶은 것만 찾아서 보여주는 거 만들어줘요”



## 7) 실패한다면? 100% C-level 때문이다

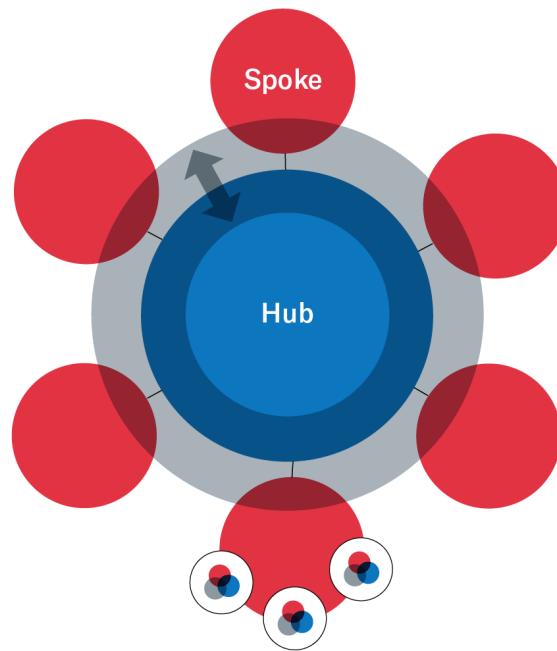
- 아주 흔한 실패 과정

- “AI대학원 나왔다며. 빨리 데이터 구축해서 모델 개발해서 서비스해야지??”
- “서비스 안 만들 거야??”
- “AI가 이것밖에 못해?”
- “맨날 안된다고만 할 거야??”
- “AI 별거 없네? 이럴 거면 뭐 하려 해”



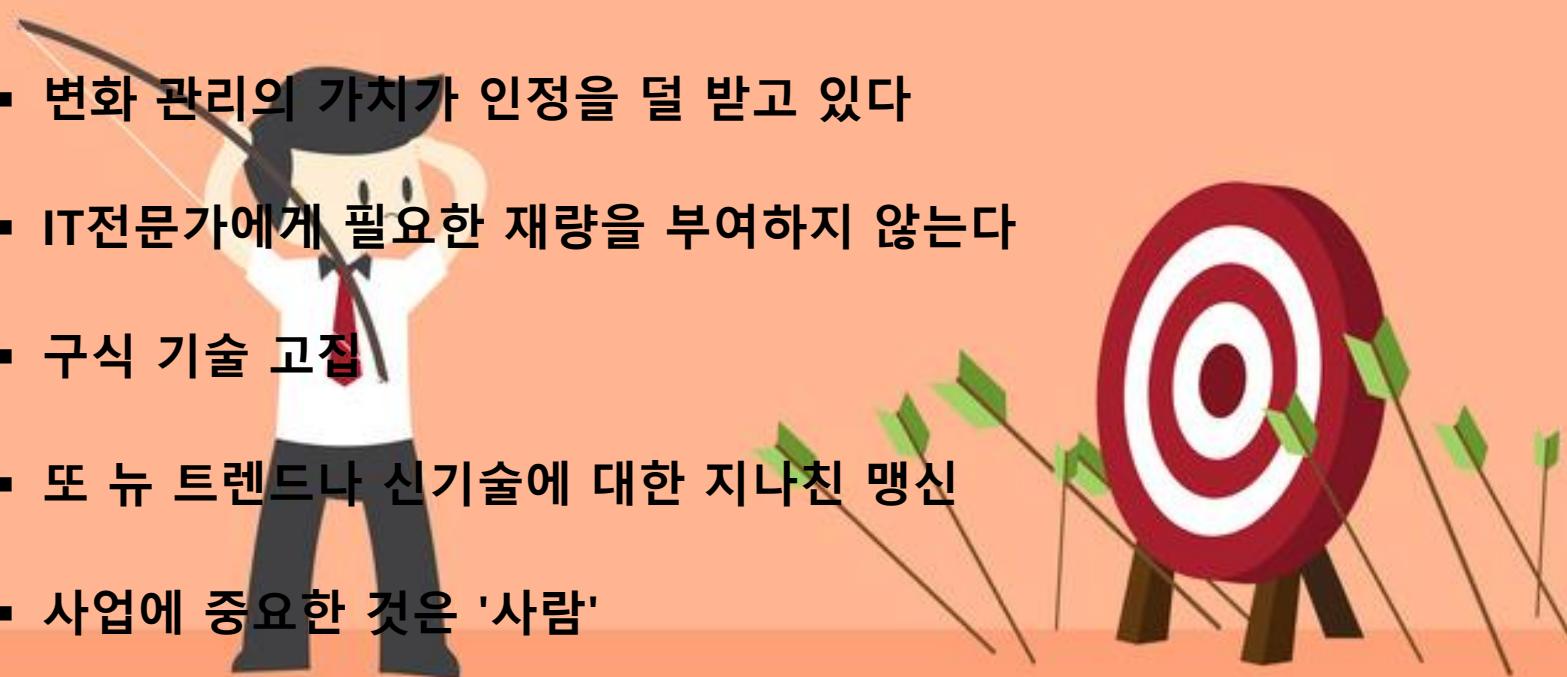
# 7) 실패한다면? 100% C-level 때문이다

- 아주 흔한 필패 과정
  - “왜 사업 부서와 협업이 안 되나?”
  - “왜 자꾸 AI 개발자들이 퇴사하지? 요즘 애들은 근성이 없나?”



## 7) 실패한다면? 100% C-level 때문이다

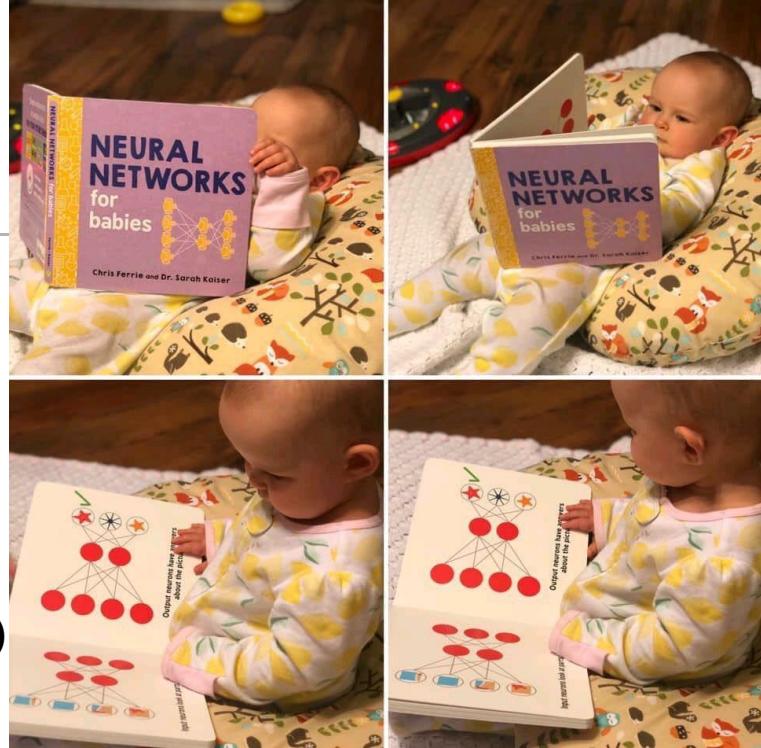
- 조직 전반에 걸쳐 합의가 부족하다
- 성공에 대한 명확한 정의가 없다
- 작업에 우선순위가 부여되지 않는다
- 변화 관리의 가치가 인정을 덜 받고 있다
- IT전문가에게 필요한 재량을 부여하지 않는다
- 구식 기술 고집
- 또 뉴 트렌드나 신기술에 대한 지나친 맹신
- 사업에 중요한 것은 '사람'



## 7) 실패한다면? 100% C-level

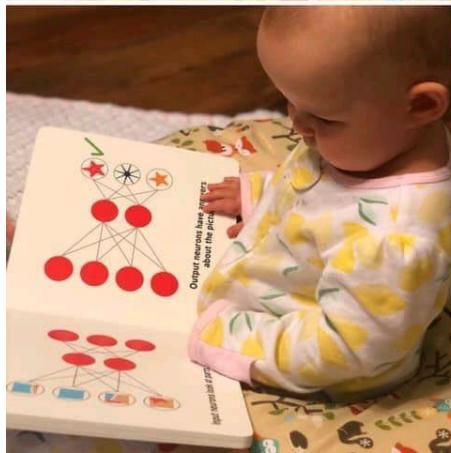
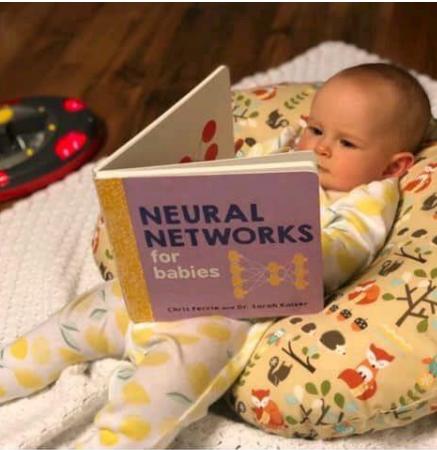
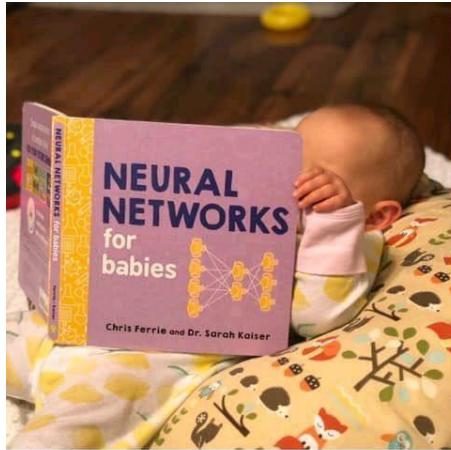
- 성공 필수 요건

- 결정권자의 AI 이해도
- 모두를 교육해야한다(임원~관리자 모두)
- 외부 컨설팅 (3회 이상)
- 충분한 인력 (데이터, AI, MLOps, ...)
- 인재 유출방지를 위한 보상
- C레벨이 직접 AI부서와 타부서 간 긴밀한 협조 이끌어내기



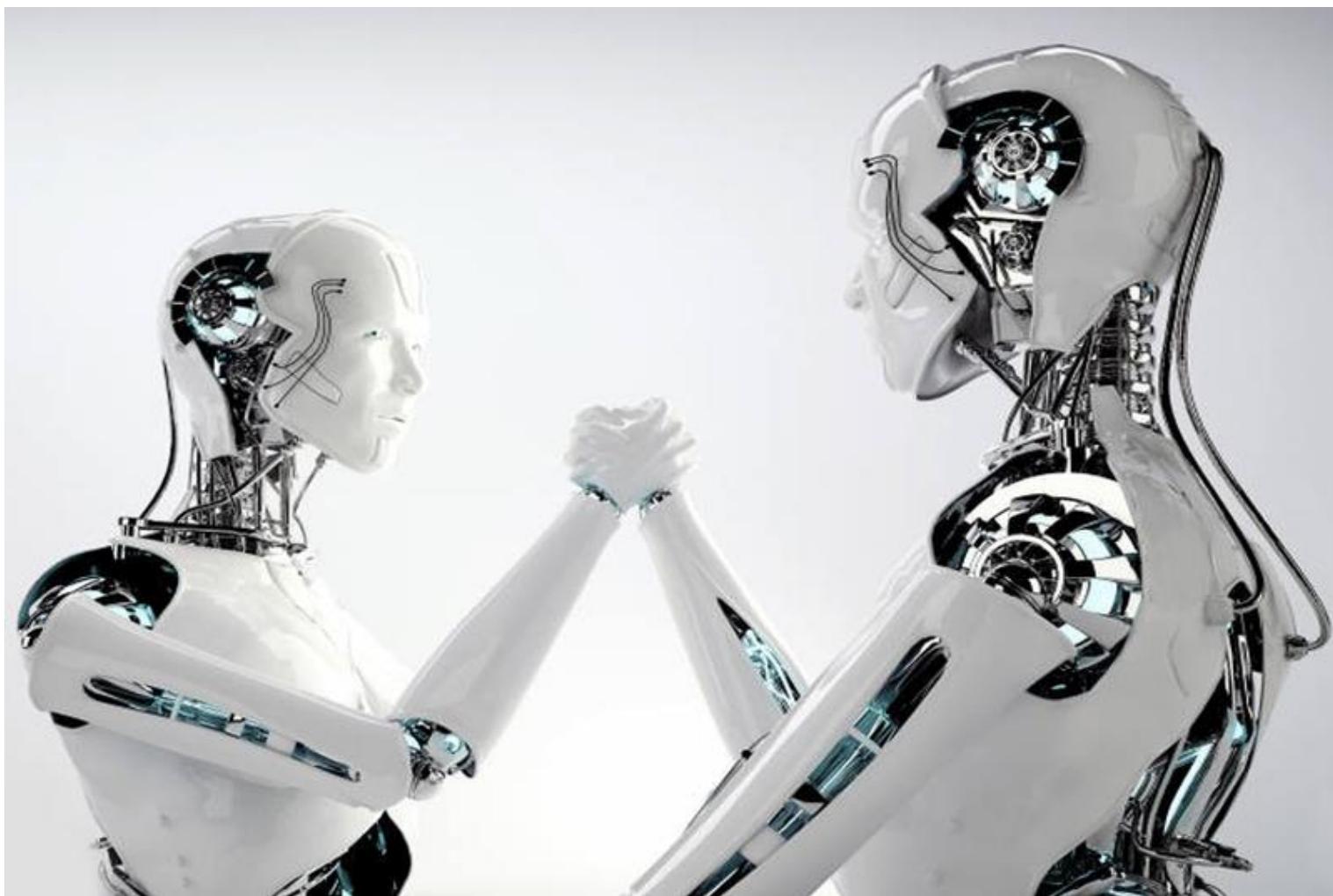
# 8) 성공하려면...

- 모두를 교육해야한다. 임원부터 관리자 직원에 까지



# What should we do?

---



# 1) NLP Intro

# Contents

---

- 언어모델

- BERT
- GPT
- ELECTRA
- MASS

- 언어모델 활용

- 개체명인식
- 문서 요약
- 번역
- QA

# 자연어처리

---

- **자연어, Natural Language**

- 우리가 일상 생활에서 사용하는 인간의 언어

- **자연어처리, Natural Language Processing(NLP)**

- 컴퓨터가 인간의 언어를 이해/모사 하도록 하는 학문
  - 이를 바탕으로 각종 정보처리
  - 언어학+컴퓨터공학+인공지능
  - for 구조화 되지 않은 비정형 텍스트 데이터

- **적용 분야**

- 음성 인식, 요약, 번역, 감성 분석, 텍스트 분류, 질의 응답 시스템, 챗봇<sup>15-235</sup> 등

# Conversational AI Technologies



Automatic Speech  
Recognition



Natural Language  
Processing



Text to  
Speech

# NLP TASKS

Machine Translation	Sentiment Analysis	Question Answering	Automatic Text Summarization
			
Author Attribution	Named Entity Recognition	Text Classification	Spell Checking
			

# 자연어처리

---

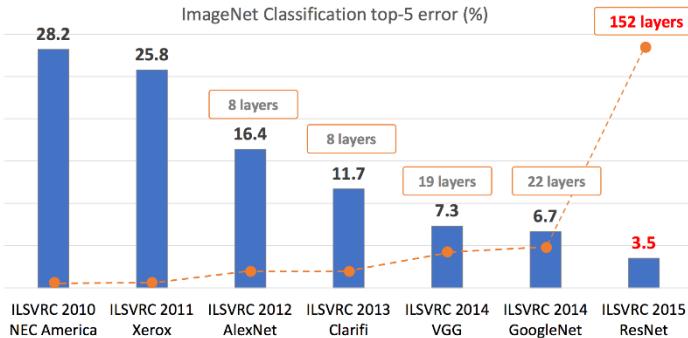
- **NLU(Natural Language Understanding)**

- 자연어 형태의 문장을 이해하는 기술
- 문서 분류, [개체명인식](#), 형태소분류

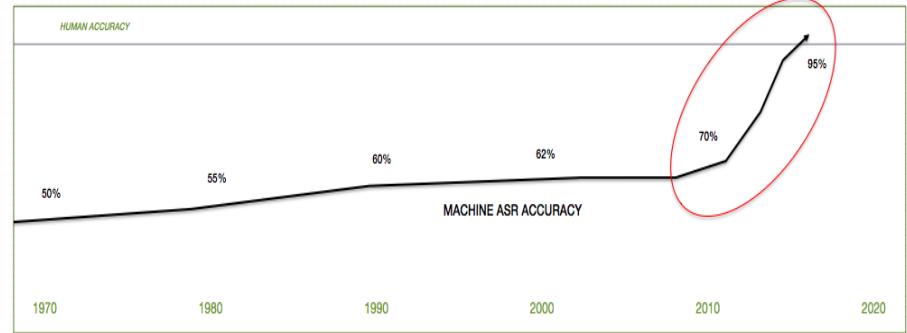
- **NLG(Natural Language Generation)**

- 자연어 문장을 생성하는 기술
- 번역, 문서 요약, 챗봇, ..

# 딥러닝



[이미지 분류]



[음성 인식]

- 2014년까지 NMT는 딥러닝으로 큰 성과 없었음
- 통계 기반 기계번역(Statistical Machine Translation, SMT)가 지배
- 규칙기반의 번역방식(Rule based Machine Translation, RBMT)
- Seq2seq를 기반으로 attention 메커니즘이 제안되며 결국 기계번역은 신경망 기계번역으로 대통합

# 자연어 처리의 어려움 1

## ■ 모호성

- 단어의 중의성 때문에 문장을 해석하는데 모호함
- 문장 내 정보의 부족으로 인한 모호성
  - 최소한의 표현으로 최대한의 정보를 표현

원문 차를 마시러 공원에 가던 차 안에서 나는 그녀에게 차였다.

Google I was kicking her in the car that went to the park for tea.

Microsoft I was a car to her, in the car I had a car and went to the park.

Naver I got dumped by her on the way to the park for tea.

Kakao I was in the car going to the park for tea and I was in her car.

SK I got dumped by her in the car that was going to the park for a cup of tea.

원문 나는 철수를 안 때렸다.

해석#1 철수는 맞았지만, 때린 사람이 나는 아니다.

해석#2 나는 누군가를 때렸지만, 그게 철수는 아니다.

해석#3 나는 누군가를 때린 적도 없고, 철수도 맞은 적이 없다.

원문 선생님은 울면서 돌아오는 우리를 위로 했다.

해석#1 (선생님은 울면서) 돌아오는 우리를 위로 했다.

해석#2 선생님은 (울면서 돌아오는 우리를) 위로 했다.

# 자연어 처리의 어려움 1

---

## ■ 모호성

- John and Henry's parents arrived at the house
  - John과 Henry의 부모님이 집에 도착했습니다
  - 총 몇명이 도착했을까요?
- 
- John, Henry, 그리고 Henry의 부모님이 도착했습니다(4명).
  - John과 Henry의 부모님이 도착했습니다(3명).
  - John과 Henry의 공통 부모님이 도착했습니다(2명).

# 자연어 처리의 어려움 2

## ■ 다양한 표현



번호  
표현

1. 골든 리트리버 한마리가 잔디밭에서 공중의 원반을 향해 달려가고 있습니다.
2. 원반이 날라가는 방향으로 개가 뛰어가고 있습니다.
3. 개가 잔디밭에서 원반을 쫓아가고 있습니다.
4. 잔디밭에서 강아지가 프리스비를 향해 뛰어가고 있습니다.
5. 높이 던져진 원반을 향해 멍멍이가 신나게 뛰어갑니다.
6. 노란 개가 원반을 잡으러 뛰어가고 있습니다.

# 자연어 처리의 어려움 3

---

- 불연속적(Discrete, Not Continuous) 데이터

- 각 단어를 discrete한 심볼로 다루었기 때문에, 마치 어휘의 크기만큼의 차원
    - 단어 임베딩을 통해서 차원 축소(dimension reduction)

- 노이즈, 정규화

- 다른 종류의 데이터에 비해서 데이터가 살짝 바뀌었을 때의 의미의 변화가 훨씬 큼

# 자연어 처리의 어려움 4 - Word Order

---

- English (Subject-Verb-Object, SVO: 42%)
  - John ate apples
- Japanese (Subject-Object-Verb, SOV: 45%)
  - Jon wa ringo o tebeta
  - John      apple      ate

# 자연어 처리의 어려움 5 - Null Subject

---

- English

- It is raining

- Japanese

- Está lloviendo
  - is raining

# 한국어 자연어처리의 어려움 1

## ▪ 교착어 (어순)

종류	대표적 언어	특징
교착어	한국어, 일본어, 몽골어	어간에 접사가 붙어 단어를 이루고 의미와 문법적 기능이 정해짐
굴절어	라틴어, 독일어, 러시아어	단어의 형태가 변함으로써 문법적 기능이 정해짐
고립어	영어, 중국어	어순에 따라 단어의 문법적 기능이 정해짐

- 접사가 붙어 같은 단어가 다양하게 생겨남
- 하나의 어근에서 생겨난 비슷한 의미의 단어가 정말 많이 생성
- 추가적인 분절을 통해서 같은 어근에서 생겨난 단어를 처리

# 한국어 자연어처리의 어려움 2

## ■ 띄어쓰기의 어려움

- 근대에 들어와서 도입
- 표준이 계속 바뀌어 왔기 때문에, 사람마다 띄어쓰기를 하는 것이 다름
- 띄어쓰기가 아예 없더라도 해석이 가능
- 추가적인 분절을 통해서 띄어쓰기를 정제(normalization) 해 주는 과정이 필요



# 한국어 자연어처리의 어려움 3

## ■ 평서문과 의문문의 차이

- 한국어에서도 의문문을 나타낼 수 있는 접사가 있음 : ~니
  - "점심 먹었니"라고 하면 굳이 물음표가 붙지 않더라도 의문문인 것을 알 수 있음
- 하지만, 많은 경우에 그냥 의문문과 평서문이 같은 형태임
  - 마침표나 물음표가 붙지 않을 경우에는 알 수가 없음

언어

평서문

의문문

---

영어

I ate my lunch.

Did you have lunch?

---

한국어

점심 먹었어.

점심 먹었어?

# 한국어 자연어처리의 어려움 4

---

## ■ 주어 생략

- 영어는 기본적으로 특성상 명사가 굉장히 중요시
- 특별한 경우를 제외하고는 주어가 생략되는 경우가 없음
- 하지만 한국어는 동사를 중요시하기 때문에, 주어가 자주 생략
- 인간은 문맥(컨텍스트) 정보를 잘 활용하여 생략된 정보를 메꿀 수 있지만, 컴퓨터는 할 수가 없음

## 2) Preprocessing

colab

preprocessing

text 분석 실습

# Colab

---

- 1) Colab 사용법: <https://youtu.be/rNgsRZ2C1Y>
- 2) Colab 실습: nlp\_2\_1\_colab.ipynb

## Colab이란?

Colaboratory(줄여서 'Colab'이라고 함)을 통해 브라우저 내에서 Python 스크립트를 작성하고 실행할 수 있습니다.

- 구성이 필요하지 않음
- GPU 무료 액세스
- 간편한 공유

학생이든, 데이터 과학자든, AI 연구원이든 Colab으로 업무를 더욱 간편하게 처리할 수 있습니다. [Colab 소개 영상](#)에서 자세한 내용을 확인하거나 아래에서 시작해 보세요.

# Preprocessing

## 1) Noise canceling

- 개행문자/특수문자/공백/중복 표현/링크/불용어

## 2) Tokenizing

- 한국어는 교착어이므로 형태소 단위 분절

## 3) POS tagging

## 4) Filtering

- 조사/띄어쓰기/문장분리

## 5) Term vector representation

## 6) Transformation TF to TF-IDF

## 7) Apply algorithm



# Preprocessing

## ■ 데이터 정제

- 특수문자 체크
    - 400개의 키보드로 입력할수 없는 특수문자 확인
  - 절반이 한자
  - 삭제 시 문서 본문의 의미에 영향을 미칠만한 특수문자가 많음
    - 단위, 주요 한자(중국, 미국), 증가/감소
    - 의미 있는 특수문자는 replace로 살리고 삭제

↑ : 457,  
→ : 439,  
↓ : 188,

```
lines = lines.replace('↑', '증가').replace('↓', '하락').replace('→', '→').replace('₩', '주식회사 ')
lines = lines.replace('外', '외신').replace('比', '대비').replace('新', '신').replace('韓', '한국').replace('史', '역사')
lines = lines.replace('金', '김').replace('女', '여자').replace('年', '년').replace('季', '이').replace('男', '남자')
# 이상한 문자들 교체
lines = lines.replace('%', '%').replace('&', '&').replace('+', '+').replace(' ', ' ').replace('-', '-')
# 로마 숫자
lines = lines.replace('Ⅰ', '1.').replace('Ⅱ', '2.').replace('Ⅲ', '3.').replace('Ⅳ', '4.').replace('Ⅴ', '5.')
lines = lines.replace('①', '1.').replace('②', '2.').replace('③', '3.').replace('④', '4.').replace('⑤', '5.')
lines = lines.replace('₩', '\u20a9').replace('₩', '\u20a9').replace('₩', '\u20a9').replace('₩', '\u20a9').replace('₩', '\u20a9')
lines = lines.replace('₩', '\u20a9').replace('₩', '\u20a9').replace('₩', '\u20a9').replace('₩', '\u20a9').replace('₩', '\u20a9')
lines = lines.replace('₩', '\u20a9').replace('₩', '\u20a9').replace('₩', '\u20a9').replace('₩', '\u20a9').replace('₩', '\u20a9')
```

# Preprocessing

# Preprocessing

```
# '~, `~, `\\~, `~, `₩' 는 유일함
lines = lines.replace('`~', '→').replace('`→', '→').replace('`→', '→').replace('`→', '→').replace('`→', '→')
lines = lines.replace('`+', '→').replace('`+', '→')
lines = lines.replace('`↑', '↑').replace('`↑', '↑').replace('`↑', '↑').replace('`↑', '↑').replace('`↑', '↑')
lines = lines.replace('`↓', '↓').replace('`↓', '↓').replace('`↓', '↓')
lines = lines.replace('`*' , '→').replace('`*' , '→').replace('`*' , '→')
lines = lines.replace('`*' , '→')
lines = lines.replace('`ß', '♂').replace('`ß', '♂').replace('`♥', '♥').replace('`♥', '♥').replace('`♥', '♥')
lines = lines.replace('`()', '') .replace('`<>', '')

# 풀임말: 온점 2개짜리 풀임말로 통일
lines = re.sub('`..', '...', lines)
lines = re.sub('`..{3,}', '...', lines) # 3개 이상은 구분자가 아니고 풀임말임 -> .. 로 처리
lines = re.sub('`..{3,}', '...', lines) # 3개 이상 ...은 .. 2개로 줄여줌

#####
## 2차 전처리 - 교체
#####
lines = lines.replace('`''', "").replace('`''', "").replace('`-`', '-').replace('`', ' ', ',').replace('`.', '.').replace('`%', '%')

lines = lines.replace('`%', '%')
#####
## 원래 문장 제거용 코드
#####
# 원래 문장이 통째로 사라지는 것인데 그럴수 없으니 해당 문자열만 삭제하기
remove_string = '\r|\x97|\uf0b1|\uf0d5|\uf0df|\uf028|\uf029|\uf02b|\uf02d|\uf03d|\uf040|\uf044|\uf05b|\uf05d|\uf
remove_string_pattern = re.compile(remove_string)
lines = remove_string_pattern.sub('', lines).strip()

print('정제 완료')
return lines
```

# Preprocessing 실습

---

- Colab 실습: `nlp_2_2_preprocessing_2_colab.ipynb`

# Text 분석 실습

---

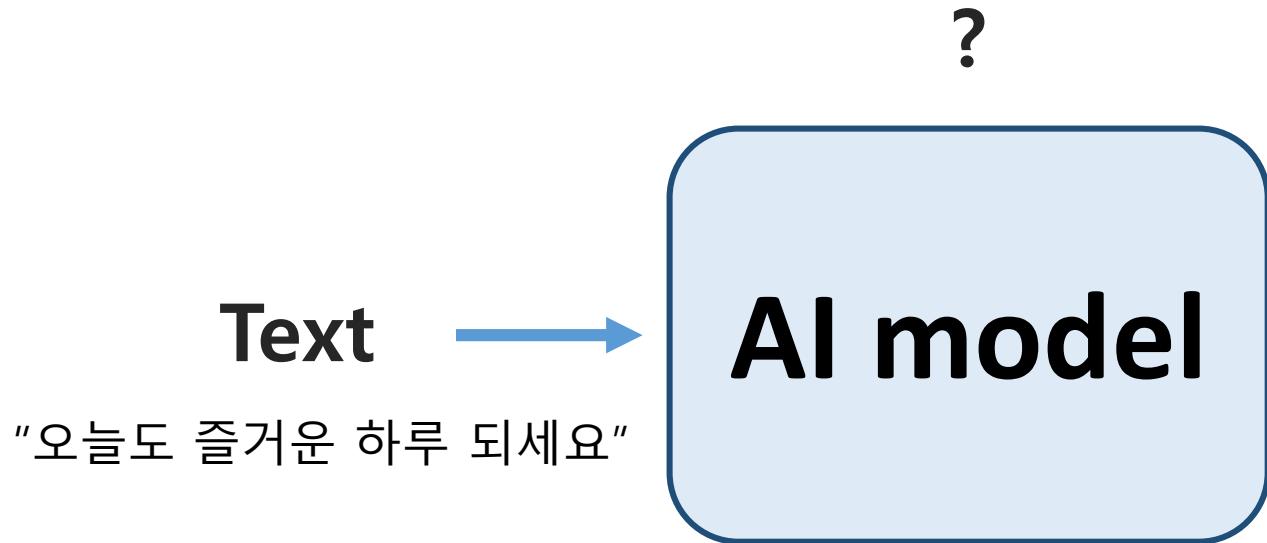
- Colab 실습: `nlp_2_3_txt_analyze_basic_3_colab.ipynb`

# 4) Traditional NLP

Text representation, LSA

# Machine Learning for Text

---



# Design Decision

?

Text



AI model

“오늘도 즐거운 하루 되세요”

Text Pre-processing

Text Representation

Reweighting

Dimension Reduction

Vector Comparison

Machine Learning Algorithm

# Design Decision

?

Text



AI model

“오늘도 즐거운 하루 되세요”

Text Pre-processing

Text Representation

Reweighting

Dimension Reduction

Vector Comparison

Machine Learning Algorithm

Word2Vec? Glove?

# Text Representations

- BOW (Bag of words)
  - 단어 단위
  - N-gram 단위
- BOW의 단점: 단어 개수만큼의 Bag이 필요함:  $|\text{Vocabulary}|$
- Let's study the NLP in the class

단어	Let's	study	the	NLP	in	class	Hard
빈도수	1	1	2	1	1	1	0

# Text Representations

- BOW (Bag of words)
- DTM (Document-Term Matrix): 문서 단어 행렬

단어	Let's	study	the	NLP	in	class	Hard
문서 1	1	1	2	1	1	1	0
문서 2	0	1	1	2	0	0	1
문서 3	1	1	0	0	2	2	1

# Text Representations

## ■ 3) TF-IDF

$$TF - IDF(w) = tf(w) \times \log \left( \frac{N}{df(w)} \right)$$

- TF (Term Frequency)
  - 특정 문서 d에서 특정 단어 t가 등장한 횟수
  - 많이 쓰인 단어가 중요하다는 가정
- DF (Document Frequency)
  - 특정 단어 t가 등장한 문서의 수
  - A라는 단어가 doc1, doc3에 등장했다면 DF는 2
  - DF가 클수록 다수 문서에 쓰이는 범용적인 단어
- IDF (Inverse Document Frequency)
  - 전체 단어수를 해당 단어의 DF로 나눈 뒤 로그를 취한 값
  - 클수록 특이한 단어

# Text Representations

## ■ 3) TF-IDF

$$TF - IDF(w) = tf(w) \times \log \left( \frac{N}{df(w)} \right)$$

- TF (Term Frequency)
  - 특정 문서 d에서 특정 단어 t가 등장한 횟수
  - 많이 쓰인 단어가 중요하다는 가정
- DF (Document Frequency)
  - 특정 단어 t가 등장한 문서의 수

	Doc1	Doc2	Doc3
Term1	5	0	0
Term2	1	0	0
Term3	5	5	5
Term4	3	3	3
Term5	3	0	1



	Doc1	TF	DF	IDF	TF-IDF
Term1	5	1	Log3	5log3	
Term2	1	1	Log3	1log3	
Term3	5	3	Log1	0	
Term4	3	3	Log1	0	
Term5	3	2	Log(3/2)	3log(3/2)	

# BOW/DTM/TF-IDF/문서검색 실습

---

- colab에서 nlp\_4\_1\_document\_vector\_2\_colab.ipynb

# BOW의 문제

---

- **Sparse representation**

- NLP: [1, 0, 0, ..., .]
- study: [0, 1, 0, ..., .]
- Word pool 사이즈에 따라 vector size가 매우 큼
- 데이터의 대부분이 0, 희소함(sparse)
- 문맥이 고려되지 않음

- **No semantic generalization**

- 의미론적 일반화 없음
- 'Cat'으로 배운 지식으로 'Dog'에 적용할 수 없음
- 관련성, 유사성 파악 불가

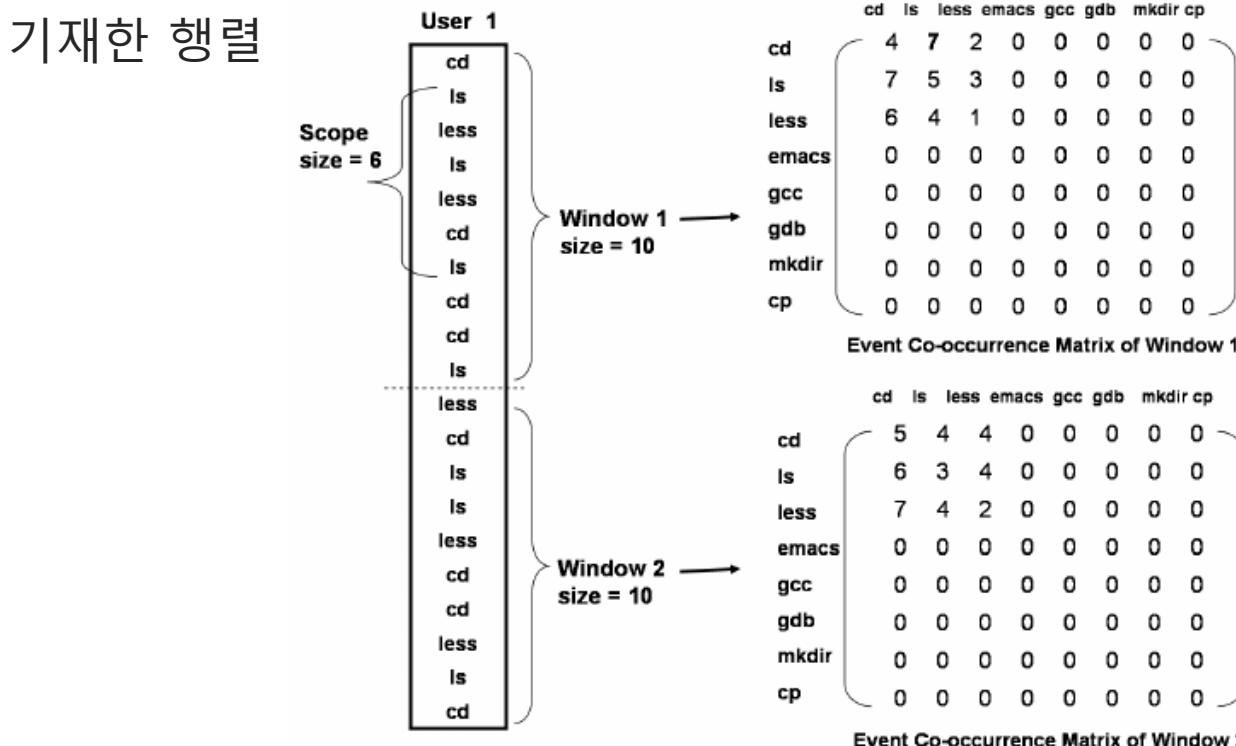
# Distribution Hypothesis

- 같은 문장에 빈번하게 같이 등장하는 단어들은 유사한 의미를 갖는다
- 유사한 단어는 유사한 분포를 갖는다
- CO-occurrence patterns

	a	big	bug	the	little	but	beetle	bit	back
a	0	5	4	2	1	0	0	3	0
big	5	0	10	8	4	0	4	8	4
bug	4	10	0	8	4	0	4	8	5
the	2	8	8	0	8	3	8	10	3
little	1	4	4	13	1	3	10	8	0
but	0	0	0	7	7	0	7	3	0
beetle	0	4	4	11	11	4	1	8	1
bit	3	8	7	12	9	3	8	0	1
back	0	4	5	3	0	0	1	2	0

# Distribution Hypothesis

- 같은 문장에 빈번하게 같이 등장하는 단어들은 유사한 의미를 갖는다
- 유사한 단어는 유사한 분포를 갖는다
- CO-occurrence patterns
- i 단어의 윈도우 크기(Window Size) 내에서 k 단어가 등장한 횟수를 i행 k열에 기재한 행렬



# Distribution Hypothesis

---

- CO-occurrence patterns

The cat sat on the mat

The dog sat on the mat

The elephant sat on the mat

The quickly sat on the mat

# Distribution Hypothesis

---

- **CO-occurrence patterns possible relationships**

- Word to documents (very sparse and very wide)
- Word to word (very dense and compact)
- Word to user / person
- Word to user behavior
- Word to product
- Word to custom feature (e.g. movie raking)
- Word to user to product

# Topic modeling

## ■ Text 문서의 의미적인 분석을 위한

### Topics

gene 0.04  
dna 0.02  
genetic 0.01  
...

life 0.02  
evolve 0.01  
organism 0.01  
...

brain 0.04  
neuron 0.02  
nerve 0.01  
...

data 0.02  
number 0.02  
computer 0.01  
...

### Documents

### Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK—How many genes does an organism need to survive? Last week at the genome meeting here,\* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

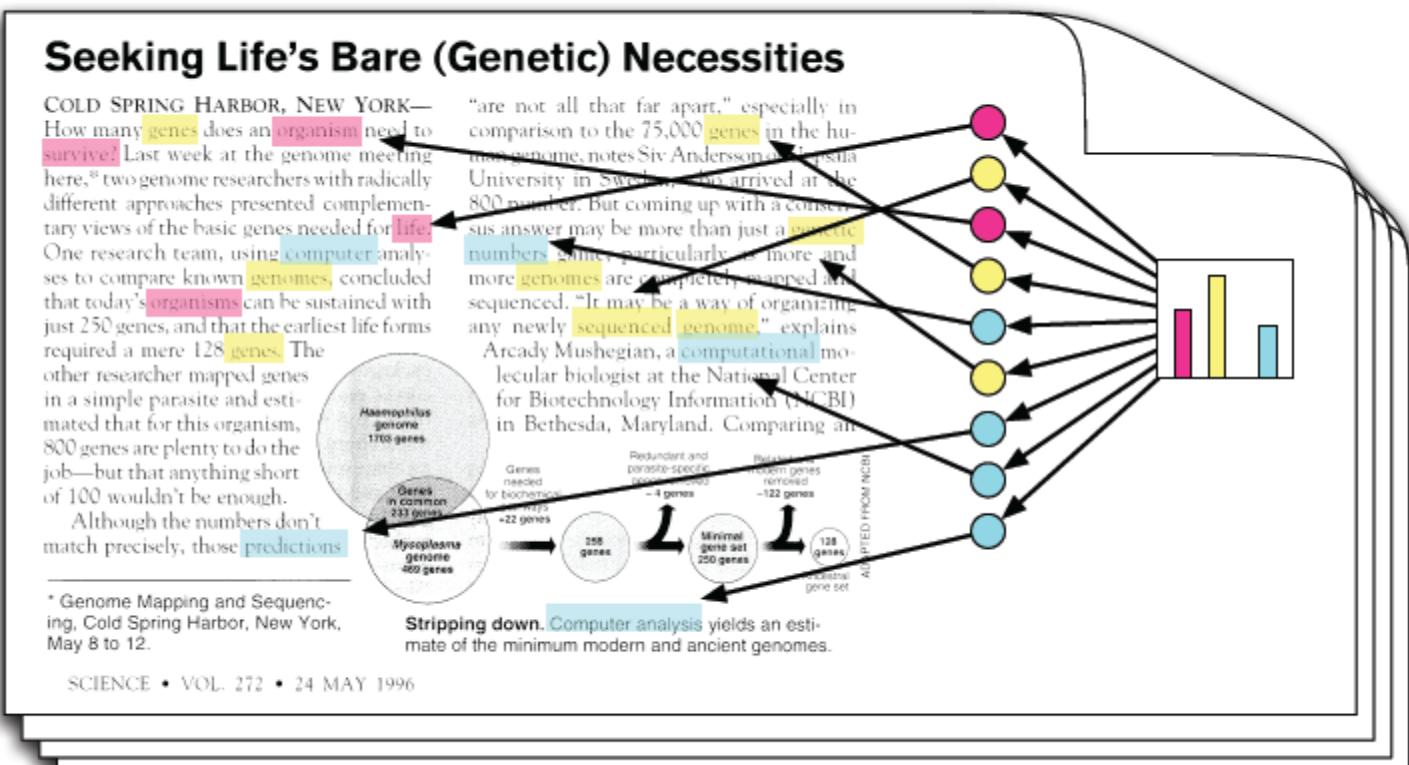
Although the numbers don't match precisely, those predictions

"are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Umeå University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

\* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

SCIENCE • VOL. 272 • 24 MAY 1996

### Topic proportions and assignments



# Topic modeling

## ■ Text 문서의 의미적인 분석을 위함

### The Starry Night

*That does not keep me from having a terrible need of — shall I say the word — religion. Then I go out at night to paint the stars. Vincent Van Gogh in a letter to his brother*

The town does not exist  
except where one black-haired tree slips  
up like a drowned woman into the hot sky.  
The town is silent. The night boils with eleven stars.  
Oh starry starry night! This is how  
I want to die.

It moves. They are all alive.  
Even the moon bulges in its orange irons  
to push children, like a god, from its eye.  
The old unseen serpent swallows up the stars.  
Oh starry starry night! This is how  
I want to die:

into that rushing beast of the night,  
sucked up by that great dragon, to split  
from my life with no flag,  
no belly,  
no cry.



### Starry Night의 요약

Starry  
Night  
Stars  
Sky  
moon

Paint  
Tree  
Orange  
Children  
flag

Exist  
Die  
Alive  
Eye  
Swallows  
life

# Topic modeling

- Text 문서의 의미적인 분석을 위함

## The Starry Night

*That does not keep me from having a terrible need of — shall I say the word — religion. Then I go out at night to paint the stars. Vincent Van Gogh in a letter to his brother*

The town does not exist  
except where one black-haired tree slips  
up like a drowned woman into the hot sky.  
The town is silent. The night boils with eleven stars.  
Oh starry starry night! This is how  
I want to die.

It moves. They are all alive.  
Even the moon bulges in its orange irons  
to push children, like a god, from its eye.  
The old unseen serpent swallows up the stars.  
Oh starry starry night! This is how  
I want to die:

into that rushing beast of the night,  
sucked up by that great dragon, to split  
from my life with no flag,  
no belly,  
no cry.



Starry Night의 요약

하늘

Starry  
Night  
Stars  
Sky  
moon

그림

Paint  
Tree  
Orange  
Children  
flag

사람

Exist  
Die  
Alive  
Eye  
Swallows  
life

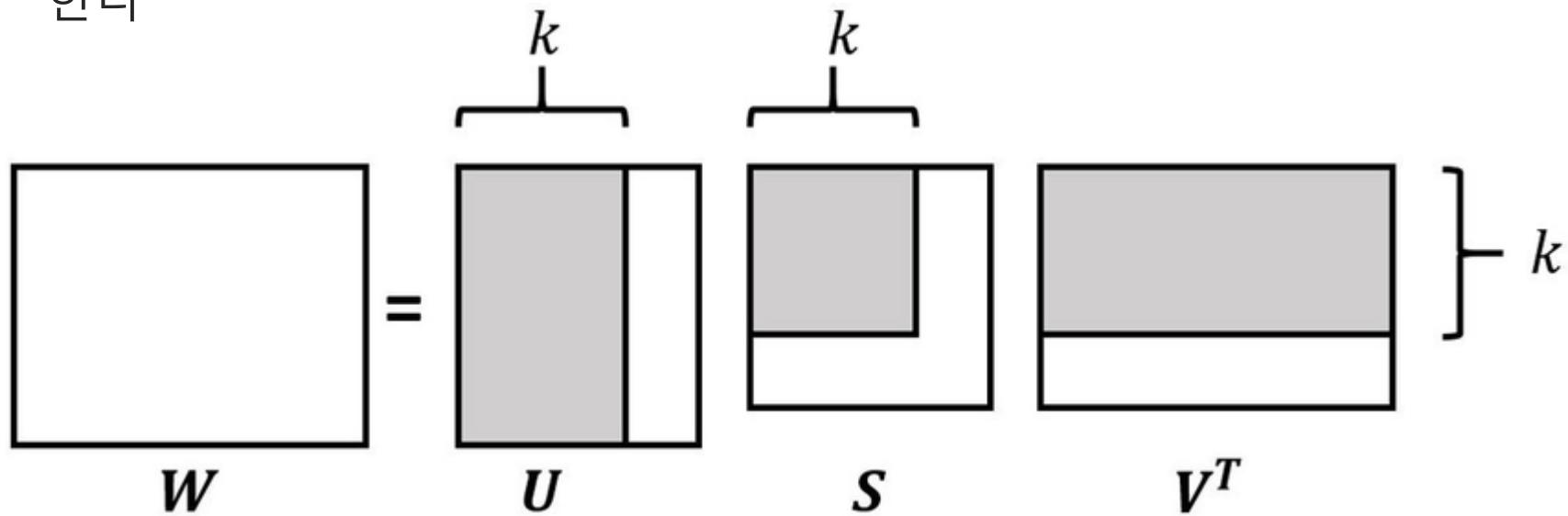
# Topic modeling

- 문서 집합의 추상적인 Topic을 발견하기 위한 통계적 모델
- 각 텍스트 본문의 숨겨진 의미 구조를 발견
- 대량의 문서들이 주어지고, term과 비슷한 문서를 찾아보자
- 의미를 탐색하기 위해 문서 내 단어들의 co-occurrence를 분석
- 문서 벡터화
  - co-occurrence로 의미적으로 유사한 문서 벡터를 가깝게 위치
  - DTM로부터 벡터의 거리를 Cosine 유사도로 구함
  - 너무 sparse한 벡터를 사용하여 정보 손실

	단어 1	단어 2	단어 3	단어 4
문서 1	자연어	처리		취업
문서 2	자연어		공부	대기업
문서 3	탕수육	짬뽕		
문서 4	탕수육		짜장	배달

# LSA

- TF-IDF로 구한 DTW 행렬을 truncated SVD 분해로  $k$ 개의 topic만 구한다



- 쉽고 빠르게 구현이 가능
- 단어의 잠재적인 의미를 이끌어낼 수 있어 문서의 유사도 계산 등에 서 좋은 성능
- 새로운 정보에 대해 업데이트가 어려움

# Wrap up

# 장비의 구성

---

- **RAM**

- 최소 32G

- **POWER**

- 최소 700W 이상, 1000W 추천

- **Cooling**

- GPU가 2개 이상이면 꼭 쿨링에 신경을, 수냉 추천

- **GPU**

- 무조건 NVIDIA 계열, Radeon은 X
  - Memory가 클수록 좋음
  - 한 PC에는 같은 종류의 무조건 GPU

# 장비의 구성

- **RAM**

- 최소 32G

- **POWER**

- 최소 700W 이상, 1000W 추천

- **Cooling**

- GPU가 2개 이상이면 꼭 쿨링에 신경을, 수냉 추천

- **GPU**

- 무조건 NVIDIA 계열, Radeon은 X
  - Memory가 클수록 좋음
  - 한 PC에는 같은 종류의 무조건 GPU

Google  
Colab

!!

# 딥러닝 프레임워크

---

- **Tensorflow?**
- **Keras?**
  - <https://keras.io/examples/>
- **Pytorch?**
  - <http://pytorch.kr/>
- **Caffe?**

# Text dataset

## ■ AI Hub



개방 데이터 ▾ 외부 데이터 ▾ 활용 사례 ▾ 개발 지원 ▾ 경진대회 ▾ 게시판 ▾

로그인

회원가입

### 개방 데이터

비전

음성/자연어

교육

국토환경

농축수산

안전

자율주행

헬스케어

인공지능 학습용 데이터  
다운로드 프로그램 설치

Windows & Mac용

> 간단 사용설명서  
> 맥용 설치&삭제 가이드

Ubuntu Ver.18.04용

> 간단 사용설명서

### 음성/자연어

음성/자연어

감성 대화 말뭉치

텍스트

오디오

음성/자연어

고객 응대 음성

텍스트

오디오

2020

음성/자연어

고서 한자 인식 OCR

이미지

텍스트

2020

음성/자연어

공공행정문서 OCR

이미지

2020

음성/자연어

기계독해

텍스트

음성/자연어

논문자료 요약

텍스트

2020

음성/자연어

다양한 형태의 한글 문자 OCR

이미지

2020

음성/자연어

도서자료 기계독해

텍스트

2020

음성/자연어

도서자료 요약

텍스트

2020

음성/자연어

명령어 음성(노인남녀)

텍스트

오디오

2020

음성/자연어

명령어 음성(소아, 유아)

텍스트

오디오

2020

음성/자연어

명령어 음성(일반남녀)

텍스트

오디오

2020

음성/자연어

문서요약 텍스트

텍스트

2020

음성/자연어

민원(콜센터) 질의-응답

텍스트

오디오

2020

음성/자연어

법률 지식베이스

텍스트

2018

음성/자연어

상담 음성

텍스트

오디오

2020

음성/자연어

생활 및 거주환경 기반 VQA

이미지

텍스트

2020

음성/자연어

소상공인 고객 주문 질의-응답 텍스트

텍스트

2020

음성/자연어

수어 영상

비디오

텍스트

2020

음성/자연어

시각정보 기반 질의응답

이미지

텍스트

2020

음성/자연어

야외 실제 촬영 한글 이미지

이미지

2020

음성/자연어

일반상식

텍스트

2017

음성/자연어

자유대화 음성(노인남녀)

텍스트

오디오

2020

음성/자연어

자유대화 음성(소아, 유아)

텍스트

오디오

2020

# Text dataset

## Tasks \_ Individual Tasks

### ■ KLUE

#### **KLUE-TC (a.k.a. YNAT) - Topic Classification**

TC is a task to predict a topic of a news headline. The topic is one of 7 categories: politics, economy, society, culture, world, IT/science, and sports.

Evaluation Metric

Macro F1

#### **KLUE-STS - Semantic Textual Similarity**

STS is a task which aims to predict the semantic similarity of two input sentences as a real value between 0 and 5. Note that we further binarized the prediction scores into two classes with a threshold score 3.0 (paraphrased or not) and evaluated with a classification metric.

Evaluation Metric

Pearson's r  
F1

#### **KLUE-NLI - Natural Language Inference**

NLI is a task to infer the relationship between a hypothesis sentence and a premise sentence. Given the premise, the model determines if the hypothesis is true (entailment), false (contradiction), or undetermined (neutral).

Evaluation Metric

Accuracy

#### **KLUE-NER - Named Entity Recognition**

NER is a task to detect the boundaries of named entities in unstructured text and to classify the types. A named entity can be of one of predefined entity types such as person, location, organization, time expressions, quantities and monetary values.

Evaluation Metric

Entity-level Macro F1  
Character-level Macro F1

#### **KLUE-RE - Relation Extraction**

RE is a task to identify semantic relations between entity pairs in a text. The relation is defined between an entity pair consisting of subject entity and object entity. The goal is then to pick an appropriate relationship between these two entities.

Evaluation Metric

Micro F1 (except no relation)  
AUPRC

#### **KLUE-DP - Dependency Parsing**

DP is a task that aims at finding relational information among words. The goal is to predict a graph structure and a dependency label of an input sentence based on the dependency grammar.

Evaluation Metric

UAS  
LAS

#### **KLUE-MRC - Machine Reading Comprehension**

MRC is a task of evaluating model that can answer a question about a given text passage. Specifically, we formulate the task as a span prediction task, where the answer is a text segment (coined as spans) in the passage.

Evaluation Metric

Exact Match  
ROUGE-W

#### **KLUE-DST (a.k.a. WoS) - Dialogue State Tracking**

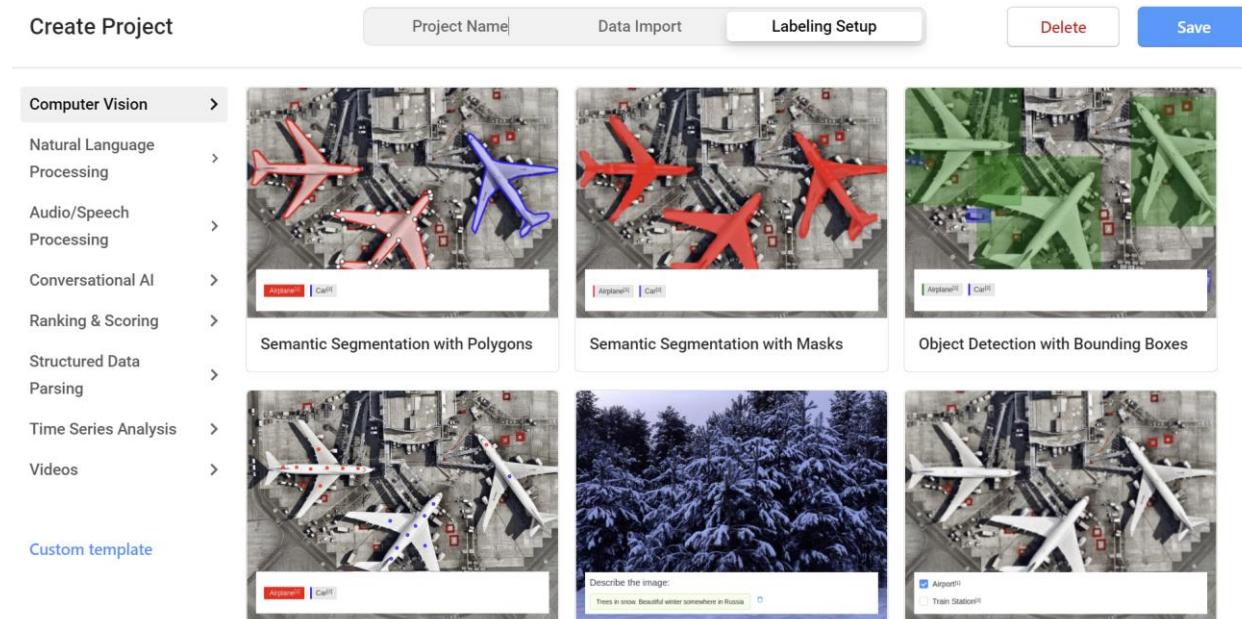
DST is a task to predict slot and value pairs (dialogue states) from a task-oriented dialogue. The potential pairs are predefined by a given task schema and knowledge base (KB).

Evaluation Metric

Joint Goal Accuracy  
Slot F1

# Annotation

- <https://labelstud.io/>
- # Install the package
  - pip install -U label-studio
- # Launch it!
  - label-studio



# Annotation

- <https://labelstud.io/>
- # Install the package
  - pip install -U label-studio
- # Launch it!
  - label-studio



```
[  
  {  
    "original_width": 600,  
    "original_height": 403,  
    "image_rotation": 0,  
    "value": {  
      "x": 24.66666666666668,  
      "y": 49.62779156327544,  
      "width": 31.66666666666668,  
      "height": 39.702233250620345,  
      "rotation": 0,  
      "rectanglelabels": [  
        "Airplane"  
      ]  
    },  
    "id": "yytSY7KW36",  
    "from_name": "label",  
    "to_name": "image",  
    "type": "rectanglelabels"  
  },  
  {  
    "original_width": 600,  
    "original_height": 403,  
    "image_rotation": 0,  
    "value": {  
      "x": 62.66666666666664,  
      "y": 65.50868486352357,  
      "width": 8.833333333333334,  
      "height": 24.317617866004962,  
      "rotation": 0,  
      "rectanglelabels": [  
        "Car"  
      ]  
    },  
    "id": "ICNn84tJme",  
    "from_name": "label",  
    "to_name": "image",  
    "type": "rectanglelabels"  
  }]  
]
```

# Word Embedding

Word2Vec, FastText

# Word embedding – Word2Vec

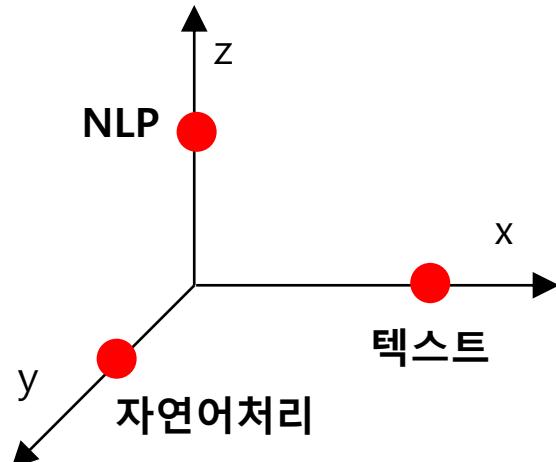
---

- 컴퓨터는 인간이 사용하는 언어를 그대로 이해하는게 아니다!
- 표현력이 무한한 언어를 벡터(숫자의 나열)로 변형하여 계산 => '**임베딩**'
- 품질 좋은 임베딩 → 좋은 NLP 성능
- 학습 수렴도 빠름
- 사람도 책을 읽을 때 기본 배경지식, 의미, 문법 정보를 가지고 독해
- 자연어를 어떻게 좌표평면 위에 표현할 수 있을까?

# Sparse Embedding

- 자연어를 어떻게 좌표평면 위에 표현할 수 있을까?

Word	Embedding
NLP	[1, 0, 0]
자연어처리	[0, 1, 0]
텍스트	[0, 0, 1]

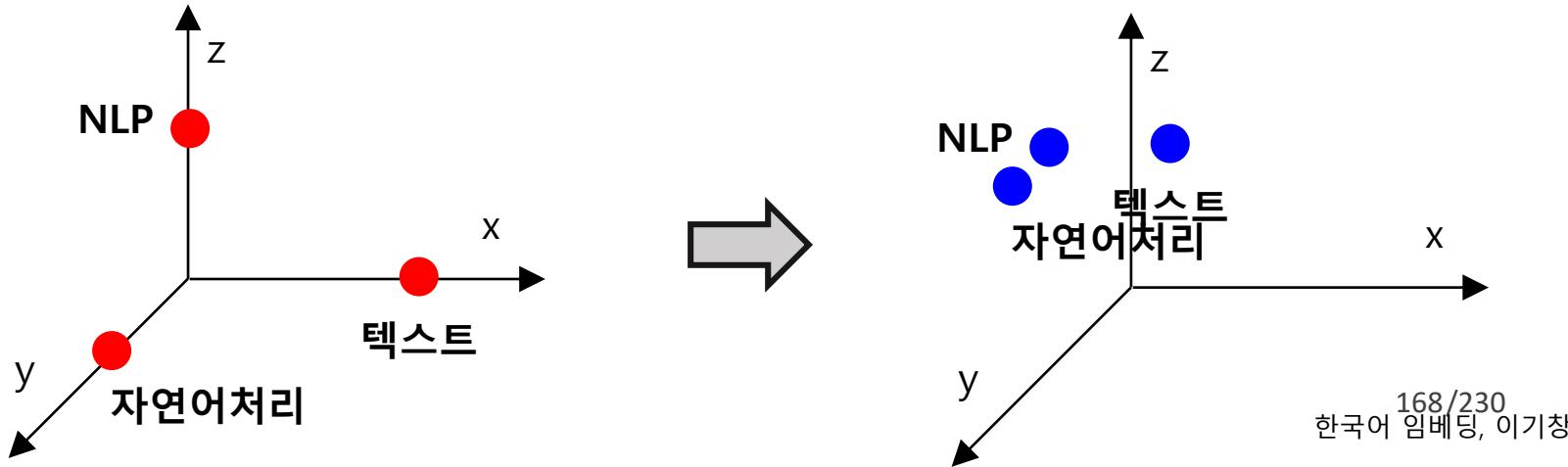


- 1) Sparse embedding

- 가장 단순한 표현 방법은 one-hot encoding 방식
- Word pool 사이즈에 따라 vector size가 매우 큼
- 데이터의 대부분이 0, 희소함(sparse)
- 문맥이 고려되지 않음
- 단어 벡터가 sparse해서 단어가 가지는 '의미'를 벡터 공간에 표현할 때, 초기화

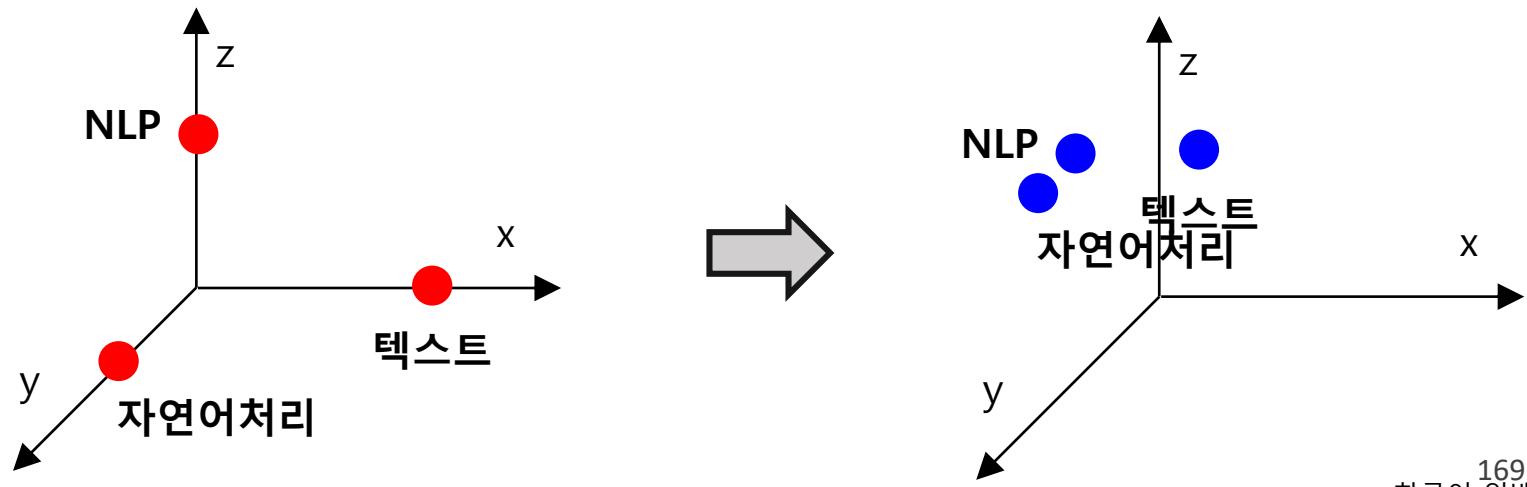
# Word2Vec

- 자연어 (특히, 단어)의 의미를 벡터 공간에 임베딩
- 한 단어의 주변 단어들을 통해, 그 단어의 의미를 파악
- ex) 비정형 데이터 처리를 위해 ★(자연어처리)를 공부해야 한다
- ex) 비정형 데이터 처리를 위해 Ⓜ(NLP)를 공부해야 한다
- ★와 Ⓜ가 무슨 뜻인지 모르겠지만, 주변 단어와 문맥을 보니 비슷한 의미 일 거 같아
- 그럼 가까이에 임베딩 해야 겠구나!



# Word2Vec

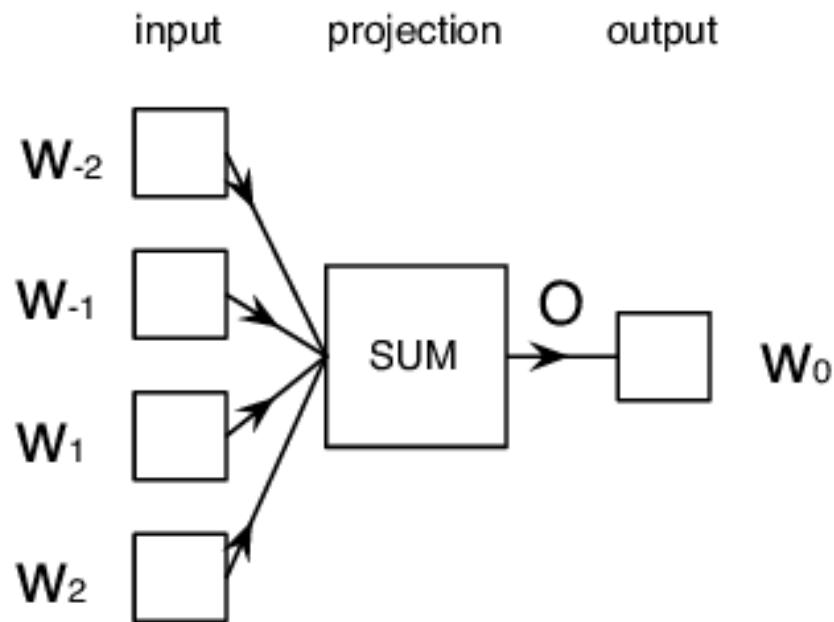
- 그런데 학습하려면 label이 있어야 하는데 어쩌지?
- 주변 단어를 보고 맞추도록 학습해보자
- **Unsupervised learning**
  - Self-supervised learning
  - 데이터셋을 스스로 생성



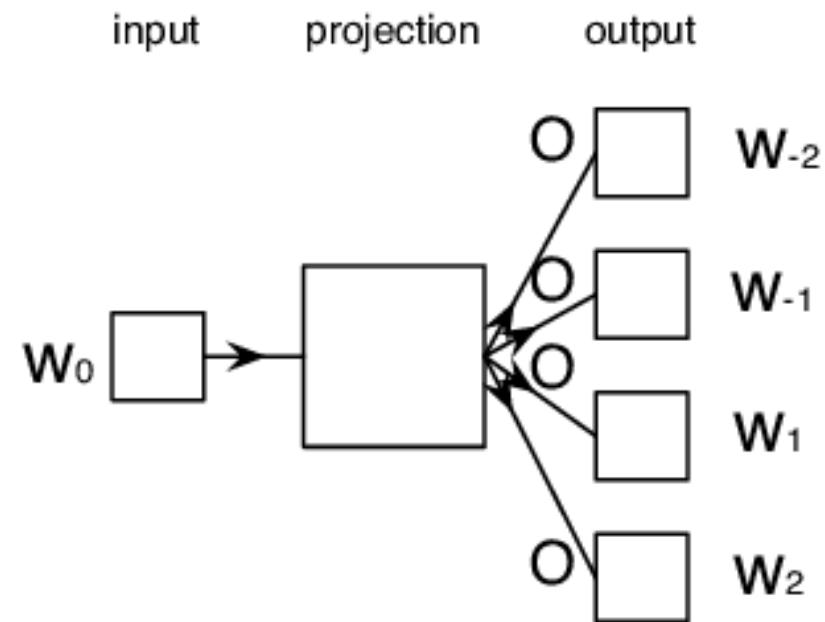
# Word2Vec

- CBOW VS Skipgram

## CBOW



## Skip-Ngram



# Word2Vec

- 1) CBOW

Source Text	Training Samples (context, target)
The quick brown fox jumps over the lazy dog. ➡	(the quick fox jumps , brown)
The quick brown fox jumps over the lazy dog. ➡	(quick brown jumps over , fox)
The quick brown fox jumps over the lazy dog. ➡	( brown fox over the , jumps)

# Word2Vec

## ■ 2) Skipgram

Source Text	Training Samples (center, target)
The quick brown fox jumps over the lazy dog. →	(the, quick) (the, brown)
The quick brown fox jumps over the lazy dog. →	(quick, the) (quick, brown) (quick, fox)
The quick brown fox jumps over the lazy dog. →	(brown, the) (brown, quick) (brown, fox) (brown, jumps)
The quick brown fox jumps over the lazy dog. →	(fox, quick) (fox, brown) (fox, jumps) (fox, over)

# Word2Vec

## ■ 2) Skipgram

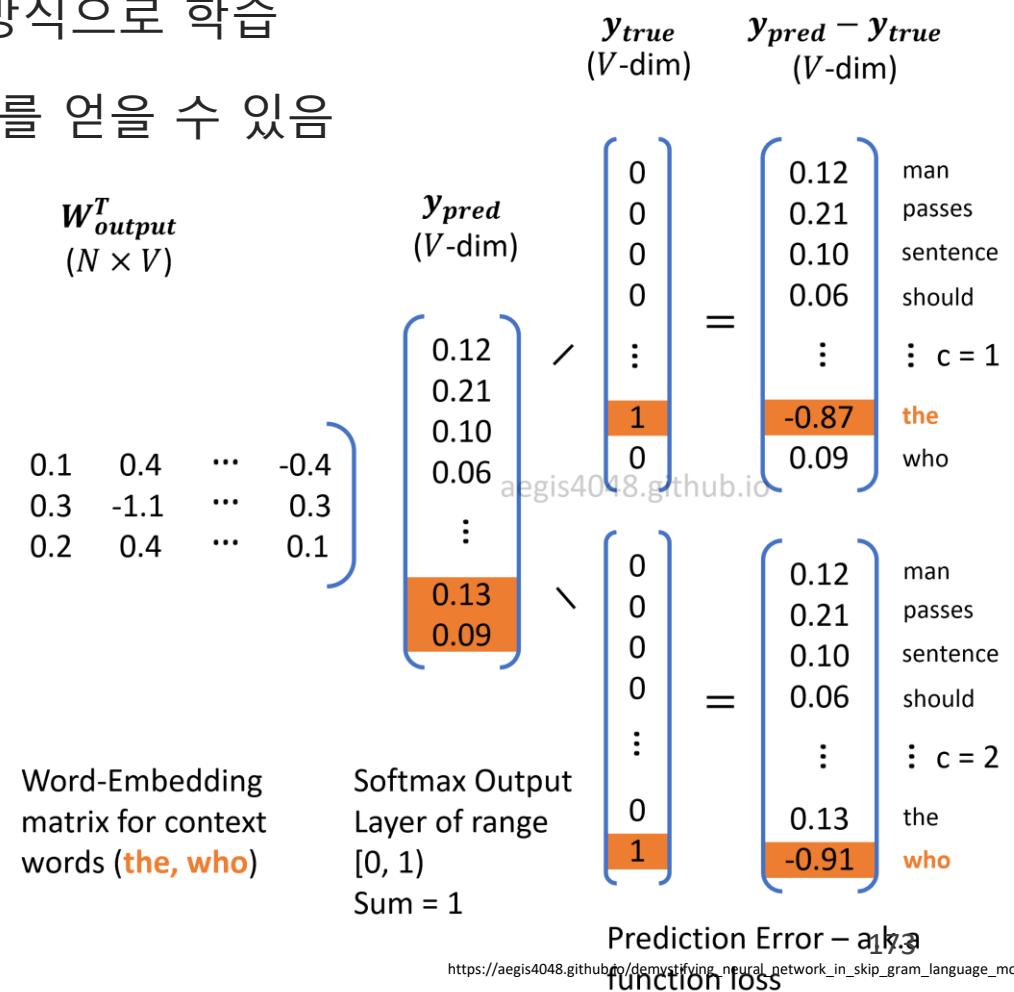
- 주변의 단어를 예측하는 방식으로 학습
- 단어에 대한 dense vector를 얻을 수 있음

	$x$ (V-dim)	$W_{input}$ ( $V \times N$ )	$h$ (N-dim)	$W_{output}^T$ ( $N \times V$ )
man	0	2 0.5 4		
passes	1	0.1 0.2 0.7		
sentence	0	0.3 -2 0.2		
should	0	-2 0.2 0.8		
:	:	:		
the	0	1 0.7 3		
who	0	3 5 0.2		

Input Layer  
one-hot  
encoded  
vector

Word-Embedding  
matrix – a.k.a  
“Lookup table”

Hidden  
(Projection)  
Layer for  
center word  
**(passes)**



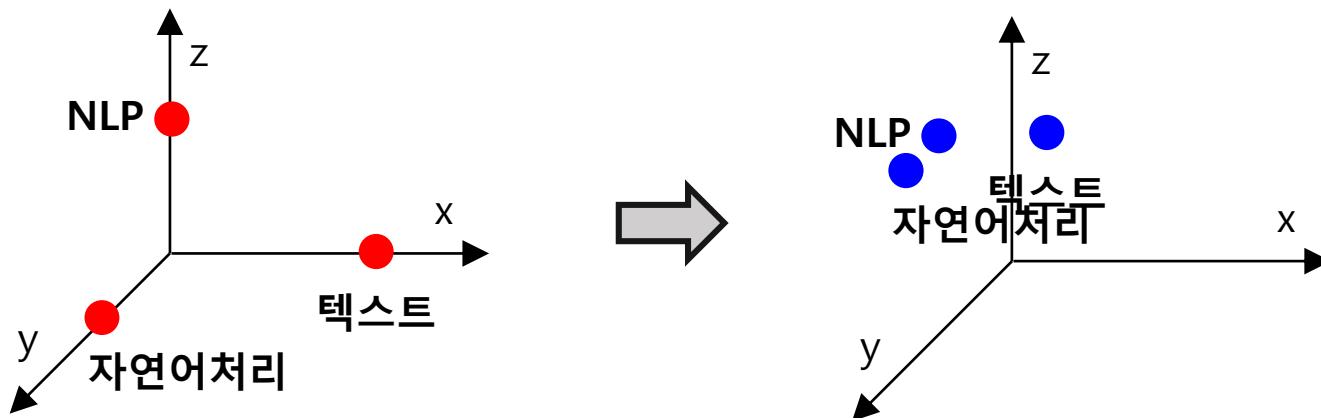
# Word2Vec

## ▪ 1) One-hot embedding

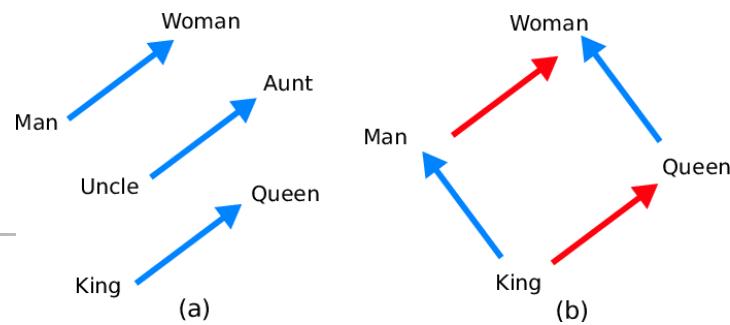
- n개의 단어에 대한 n차원의 벡터
- 단어가 커질 수록 무한대 차원의 벡터가 생성
- 의미 유추 불가능
- 차원의 저주 (curse of dimensionality): 차원이 무한대로 커지면 정보 추출이 어려워짐

## ▪ 2) Word embedding

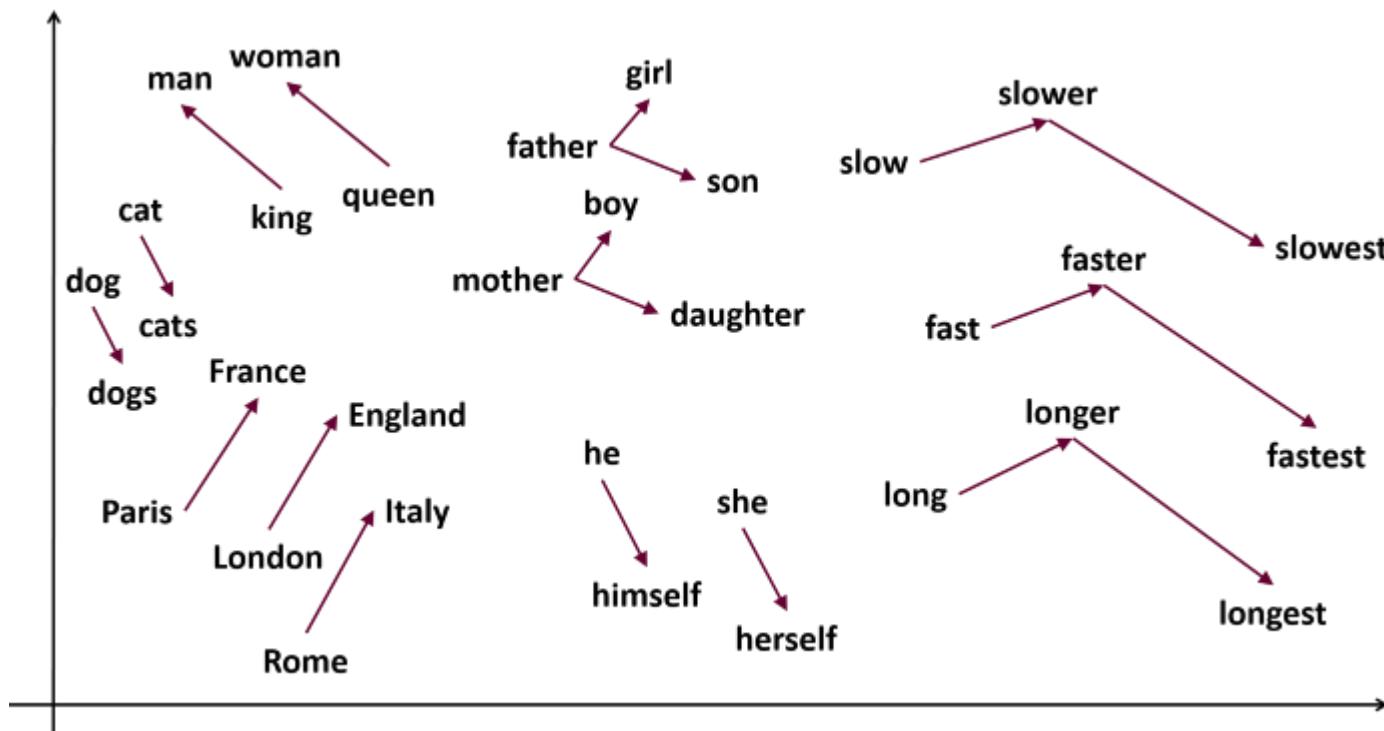
- 한정된 차원으로 표현 가능 (100d, 200d, 768d, ..)
- 의미 관계 유추 가능
- 비지도 학습으로 단어의 의미 학습 가능



# Word2Vec 벡터 연산

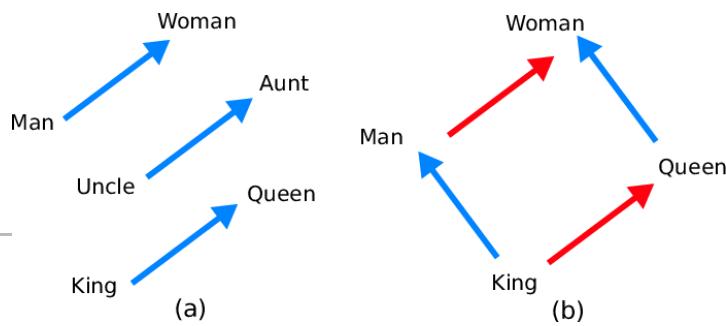


- 단어의 의미를 고려한 벡터로 표현하므로 벡터 연산이 가능
- $\overrightarrow{w_{king}} - \overrightarrow{w_{man}} + \overrightarrow{w_{woman}} \approx \overrightarrow{w_{queen}}$



# Word2Vec 장/단점

- 단어의 의미를 고려한 다차원 벡터로 표현



## ▪ 장점

- 단어간의 유사도/관계 파악에 용이
- 벡터 연산을 통한 추론 가능

## ▪ 단점

- 단어의 subword information 무시
  - ex) 대전 VS 대전시 VS 대전시청 VS 서울시청
- Out of vocabulary (OOV) 문제

# 어간/어미

---

- 어간/어미
  - 어간(stem): 용언의 활용 시 변하지 않는 부분, (~=식물의 줄기)
  - 어미(ending): 용언의 활용시 어간 뒤에 붙어서 변하는 부분
- 1) 규칙 활용: 어간이 어미를 취할 때, 어간과 어미의 형태가 둘다 변하지 않거나 규칙적으로 변하는 활용
  - 가(다) + 니 -> 가니
  - 먹(다) + 고 -> 먹고
  - 울- + 니 -> 우니(르탈락)
  - 치르(다) + 어 -> 치러(―탈락)
  - 쓰(다) + 어 -> 써(―탈락)

# 어간/어미

---

- 2) 불규칙 활용: 어간이나 어미의 모습이 변하는 활용
  - 2.1) 어간이 변하는 경우
    - 굿(다) + 어 -> 그어
    - 걷(다) + 어 -> 걸어
    - 흐르(다) + 어 -> 흘러
  - 2.2) 어미가 변하는 경우
    - 하(다) + 어 -> 하여
  - 2.3) 어간+어미 모두 변하는 경우
    - 파랗(다) + 아 -> 파래

# Word2Vec 벡터 연산

---

- ex 1) 동사: \*\*\*먹\*\*\*다, \*\*\*먹\*\*\*고, \*\*\*먹\*\*\*니, \*\*\*먹\*\*\*지, \*\*\*먹\*\*\*으며, \*\*\*먹\*\*\*어서
- ex 2) 형용사: \*\*\*예쁘\*\*\*다, \*\*\*예쁘\*\*\*고, \*\*\*예쁘\*\*\*니, \*\*\*예쁘\*\*\*지
- Stemming: 어절에서 어간 추출
  - 용언에서 어미를 제외하고 어간을 추출
    - ex) 'going' -> 'go'
    - ex) 'Computers' -> 'Comput'

# Fasttext

---

- by Facebook research
- Word2Vec의 단점을 보완
  - 단어의 subword information 무시
    - ex) 대전 VS 대전시 VS 대전시청 VS 서울시청
  - Out of vocabulary (OOV) 문제

# Fasttext

---

## ■ Training

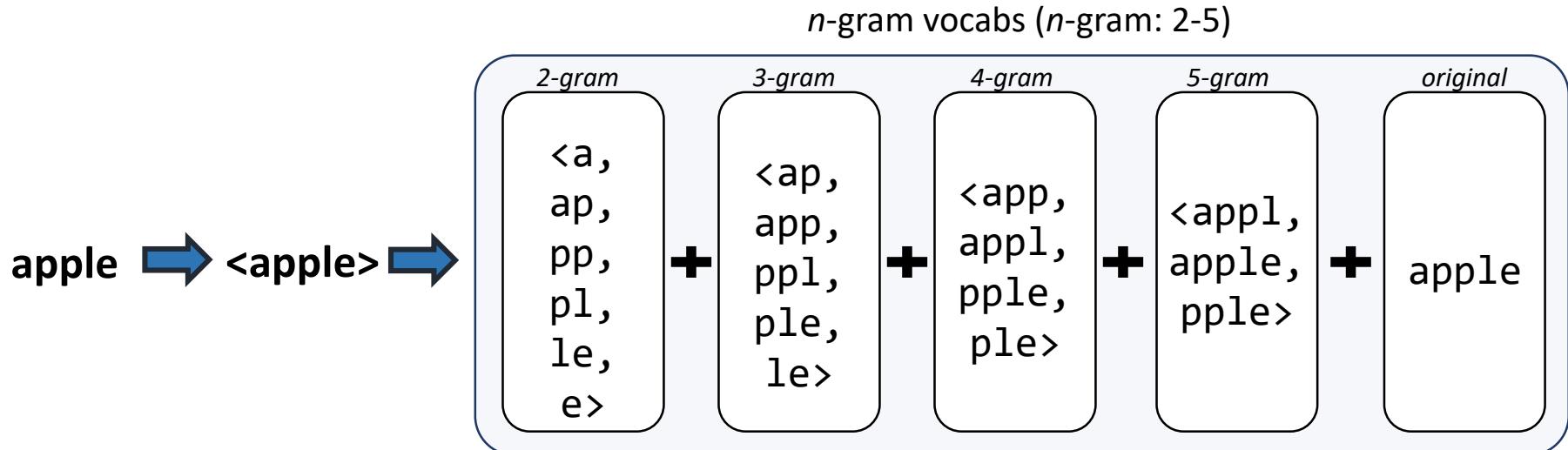
- 기존의 word2vec과 유사, 단어를 n-gram으로 나누어 학습
- n-gram의 범위가 2~5일 때, 단어를 다음과 같이 분리하여 학습함
  - “assumption” = {as, ss, su, ...., ass, ssu, sum, ump, mpt, ...., ption, assumption}
- 이 때, n-gram으로 나눠진 단어는 사전에 들어가지 않으며, 별도의 n-gram vector를 형성

## ■ Testing

- 입력 단어가 vocabulary에 있을 경우, word2vec과 마찬가지로 해당 단어의 word vector를 return
- 만약 OOV일 경우, 입력 단어의 n-gram vector들의 합산을 return

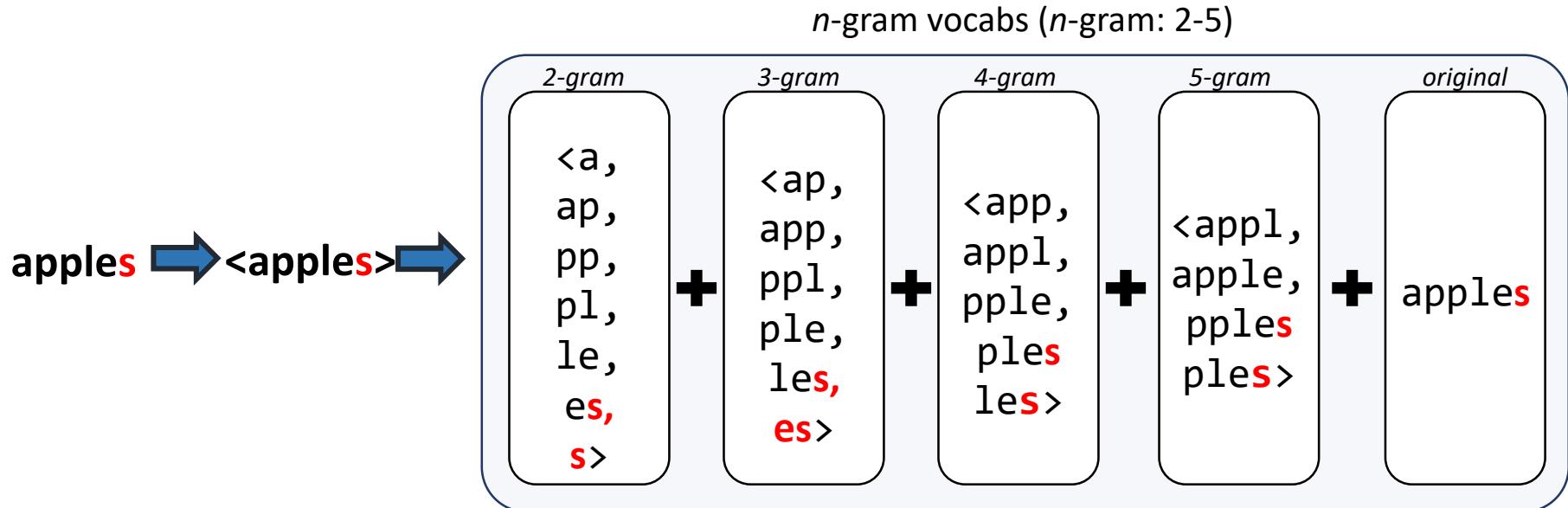
# Fasttext

- 단어를 n-gram으로 분리를 한 후, 모든 n-gram vector를 합산한 후 평균을 통해 단어 벡터를 획득



# Fasttext

- 단어를 n-gram으로 분리를 한 후, 모든 n-gram vector를 합산한 후 평균을 통해 단어 벡터를 획득



# Word2Vec & Fasttext 실습

---

- colab에서 nlp\_6\_1\_word2vec\_1\_colab.ipynb 실행

# **Subword Tokenization**

Sentencepiece, Huggingface

# Subword Tokenization

---

- 문장 내 단어들은 여러 단어가 합쳐져 쓰여짐
  - "I'd like to study NLP"
  - ", I, 'd, like, to, study, NLP, ., "
- 한국어는 어간+어미로 단어가 이루어짐
  - 한 어간에서 다양한 어미와 결합하여 여러 단어가 생성됨
  - 어간과 어미의 분리가 필요
- ex 1) 동사: **먹**다, **먹**고, **먹**니, **먹**지, **먹**으며, **먹**어서
- ex 2) 형용사: **예쁘**다, **예쁘**고, **예쁘**니, **예쁘**지

# OoV (Out-of-Vocabulary)

- 복합명사/어간+어미 처럼 여러 단어의 결합인 경우 분리하지 않으면 Vocab에 없을 확률이 높음
- OoV가 입력될 경우 '[UNK]' 토큰으로 치환하여 사용
  - 비정형 데이터 처리를 위해 를 공부해야 한다
  - -> 비정형 데이터 처리를 위해 [UNK]를 공부해야 한다
- OOV는 NLG task에서 특히 치명적인 성능 하락을 가져옴
  - 비정형 데이터 처리를 위해 [UNK]를 공부해야 한다
    - -> We have to study [UNK], [UNK], [UNK], [UNK], ...
- OOV지만 단어의 subword 정보로 어느정도 의미 파악 가능
  - ex) 대전 VS 대전시 VS 대전시청 VS 서울시청

# Subword tokenization

---

- 단어를 글자 단위로 쪼개어 자주 등장하는 글자들끼리 묶어서 subword 단위로 처리하자
- **Subword: 단어보다 작은 단위**
  - Subword -> 'Sub' + 'word'
- **Subword 알고리즘 종류**
  - Byte pair encoding,
  - CharBPE
  - ByteLevelBPE
  - WordPiece
  - SentencePiece

# BPE 학습

---

## ■ Training

- 1) 띄어쓰기 단위로 단어 사전 생성
- 2) 각 단어를 글자단위로 분절한 후 자주 함께 등장하는 pair별로 빈도수 계산
- 3) 빈도수가 높은 pair들은 merge 수행
- 반복!!

## ■ Inference

- 띄어쓰기 단위로 단어 사전 생성
- 각 단어를 글자단위로 분절
- Subword에 존재하는 pair 순서대로 merge 수행

# BPE 학습

- 입력: '경찰청 철창살은 외철창살이고, 검찰청 철창살은 쌍철창살이다.'
- 1) 띄어쓰기 단위로 단어 사전 생성
  - -> ['경찰청', '철창살은', '외철창살이고', '검찰청', '철창살은', '쌍철창살이다']

Sentence	Tokenized Sentence (iter=0)		
경찰청	→경 찰 청	→	경 ##찰 ##청
철창살은	→철 창 살 은	→	철 ##창 ##살 ##은
외철창살이고	→외 철 창 살 이 고	→	외 ##철 ##창 ##살 ##이 ##고
검찰청	→검 찰 청	→	검 ##찰 ##청
철창살은	→철 창 살 은	→	철 ##창 ##살 ##은
쌍철창살이다	→쌍 철 창 살 이 다	→	쌍 ##철 ##창 ##살 ##이 ##다

# BPE 학습

- 입력: '경찰청 철창살은 외철창살이고, 검찰청 철창살은 쌍철창살이다.'
- 2) 각 단어를 글자단위로 분절한 후 자주 함께 등장하는 pair별로 빈도수

Tokenized Sentence (iter=0)	Bi-gram pairs (iter=1)
경 ##찰 ##청	→ (경, ##찰), (##찰, ##청)
철 ##창 ##살 ##은	→ (철, ##창), (##창, ##살), (##살, ##은)
외 ##철 ##창 ##살 ##이 ##고	→ (외 ##철), (##철, ##창), (##창, ##살), (##살, ##이), (##이, ##고)
검 ##찰 ##청	→ (검, ##찰), (##찰, ##청)
철 ##창 ##살 ##은	→ (철, ##창), (##창, ##살), (##살, ##은),
쌍 ##철 ##창 ##살 ##이 ##다	→ (쌍 ##철), (##철, ##창), (##창, ##살), (##살, ##이), (##이, ##다)

(경, ##찰)	:1	(##살, ##은)	:2	(##이, ##고)	:1
(##찰, ##청)	:2	(외 ##철)	:1	(검, ##찰)	:1
(철, ##창)	:2	(##철, ##창)	:2	(쌍 ##철)	:1
(##창, ##살)	:4	(##살, ##이)	:2	(##이, ##다)	:1

# BPE 학습

- 입력: '경찰청 철창살은 외철창살이고, 검찰청 철창살은 쌍철창살이다.'
  - 3) 빈도수가 높은 pair들은 merge 수행

Best Pair: ##창 ##살 → **##창살**

Tokenized Sentence (iter=0)

경 ##찰 ##청  
철 ##창 ##살 ##은  
외 ##철 ##창 ##살 ##이 ##고  
검 ##찰 ##청  
철 ##창 ##살 ##은  
쌍 ##철 ##창 ##살 ##이 ##다

Tokenized Sentence (iter=1)

→ 경 ##찰 ##청  
→ 철 **##창살** ##은  
→ 외 ##철 **##창살** ##이 ##고  
→ 검 ##찰 ##청  
→ 철 **##창살** ##은  
→ 쌍 ##철 **##창살** ##이 ##다

# ▪ Subword 알고리즘 종류

---

## ▪ Byte pair encoding (BPE)

- 빈도수가 큰 음절 pair를 merge

- CharBPE

- ByteLevelBPE

## ▪ WordPiece

- 코퍼스의 likelihood를 가장 높이는 pari에 merge 수행
- merge가 가능한 후보군들을 서로 비교하며 merge할 가치가 있는 대상만을 merge
- {'철##창'}과 {'##창살'}
- $P(\#\#\#창살) / (P(\#\#\#창) * P(\#\#\#살))$  과  $P(\#\#\#창) / (P(\#\#\#창) * P(\#\#\#창))$

# Subword 장점

---

- 단어를 Subword로 부절하므로 OoV 문제 극복
- 형태소 분석기로 분절 후 학습하면 Subword 알고리즘이 형태소 단위 분절도 학습

# Subword tokenization 실습

---

- colab에서 nlp\_7\_1\_subword\_huggingface\_1\_colab.ipynb
- colab에서 nlp\_7\_2\_subword\_sentencepiece\_VS\_huggingface\_colab.ipynb

# Pytorch 101

실습

220527\_NLP\_pytorch.pdf

# **Convolutional Neural Network for Text Classification**

# Review of Deep Learning (CNN)

---

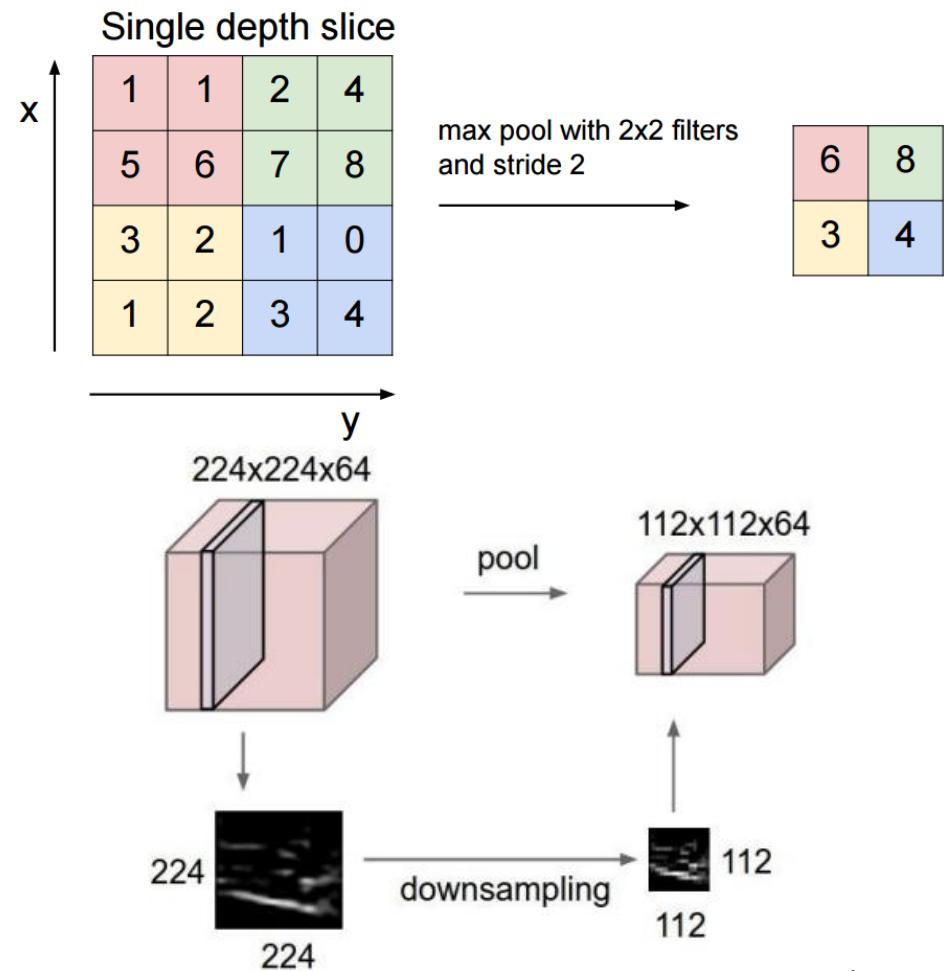
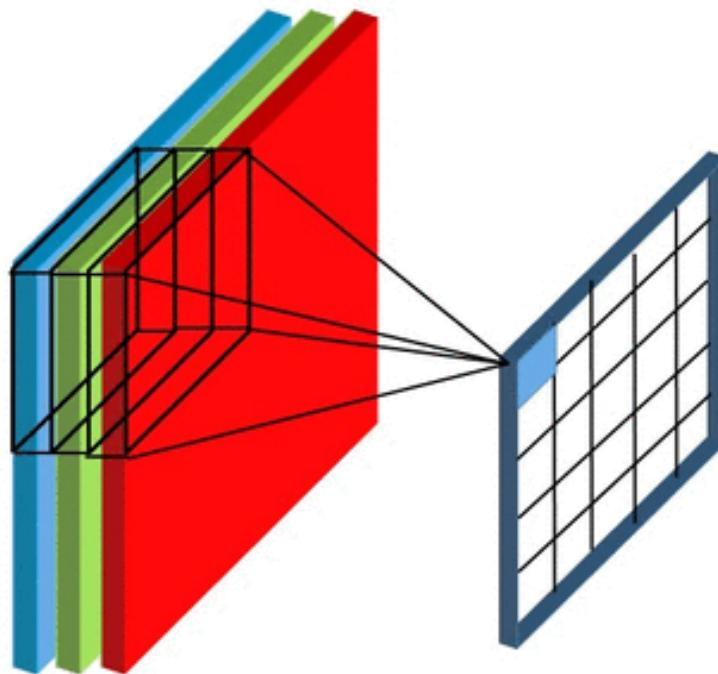
## ▪ CNN 실습

- cnn\_1\_기초.ipynb
- cnn\_3\_(숙제)CNN\_기초\_모델만들기\_실습\_정답.ipynb
- cnn\_4\_text\_CNN\_기초\_colab.ipynb

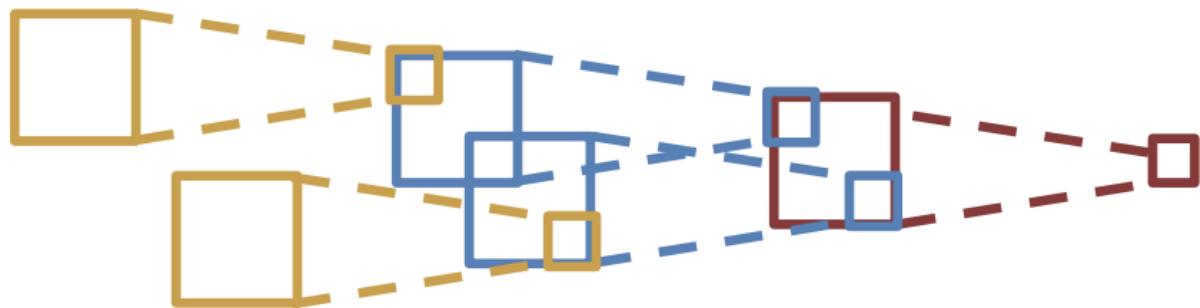
# 1D-CNN for NLP

## ▪ Convolutional Neural Networks for Sentence Classification

- 1D CNN + MaxPooling

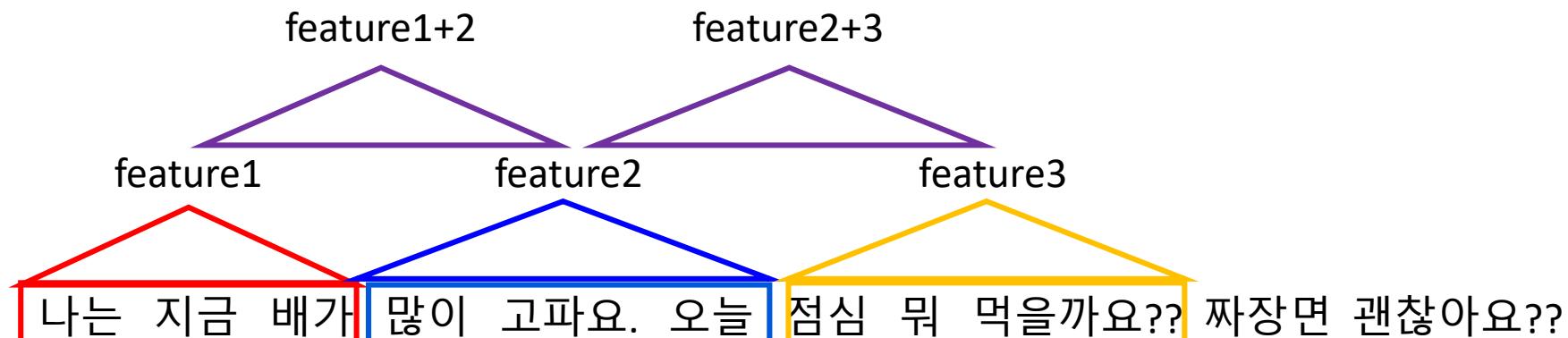


# CNN



## ▪ Receptive Fields

- k size convolution, output depends on a  $K \times K$  receptive field in the input
- Problem: For large images we need many layers for each output to "see" the whole image
- Solution: Downsample inside the network
- 문장을 분류 할 때, 특정 단어의 조합을 특징으로 추출한다면??



# 1D-CNN for NLP

## ▪ Convolutional Neural Networks for Sentence Classification

### Convolutional Neural Networks for Sentence Classification

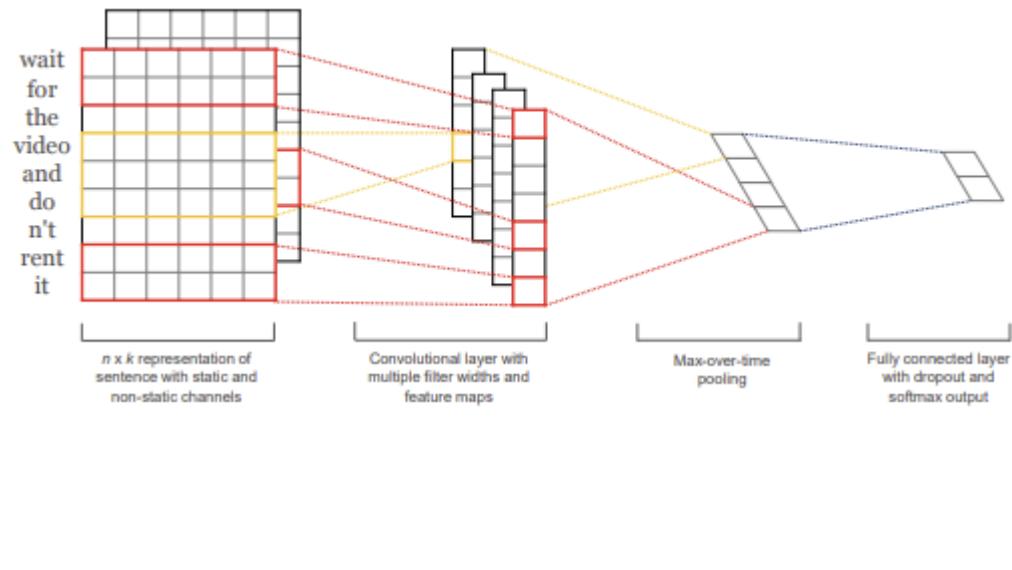
Yoon Kim  
New York University  
yhk255@nyu.edu

#### Abstract

We report on a series of experiments with convolutional neural networks (CNN) trained on top of pre-trained word vectors for sentence-level classification tasks. We show that a simple CNN with little hyperparameter tuning and static vectors achieves excellent results on multiple benchmarks. Learning task-specific vectors through fine-tuning offers further gains in performance. We additionally propose a simple modification to the architecture to allow for the use of both task-specific and static vectors. The CNN models discussed herein improve upon the state of the art on 4 out of 7 tasks, which include sentiment analysis and question classification.

local features (LeCun et al., 1998). Originally invented for computer vision, CNN models have subsequently been shown to be effective for NLP and have achieved excellent results in semantic parsing (Yih et al., 2014), search query retrieval (Shen et al., 2014), sentence modeling (Kalchbrenner et al., 2014), and other traditional NLP tasks (Collobert et al., 2011).

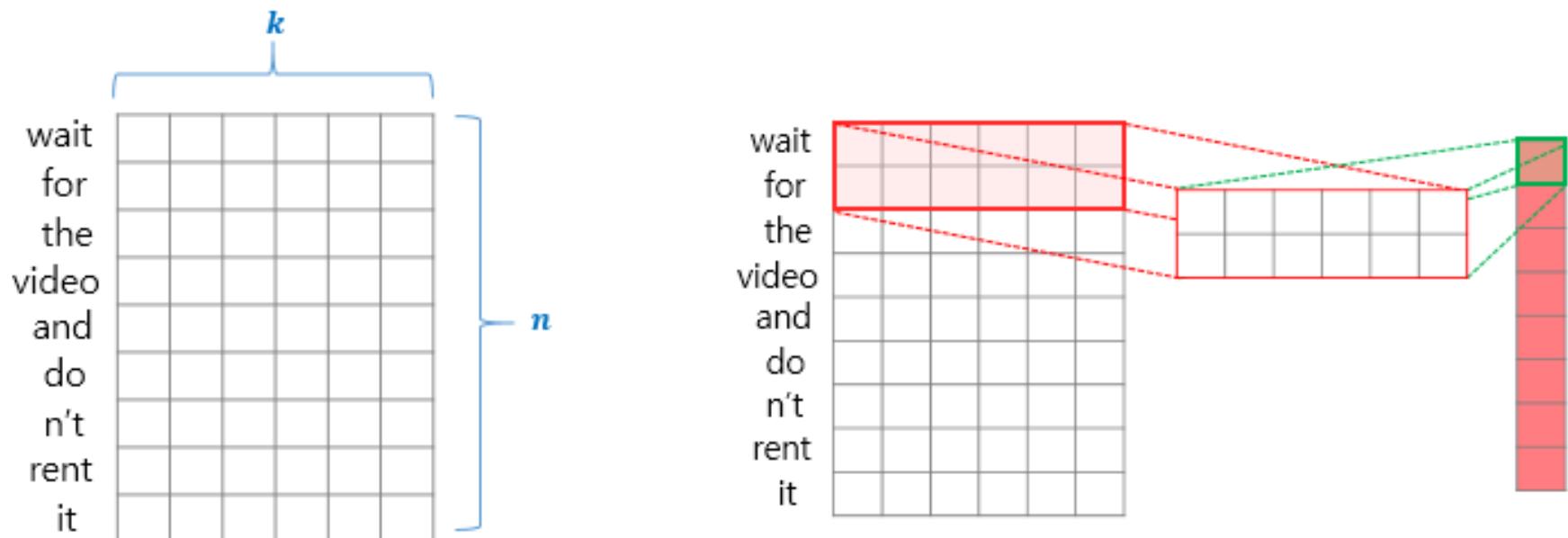
In the present work, we train a simple CNN with one layer of convolution on top of word vectors obtained from an unsupervised neural language model. These vectors were trained by Mikolov et al. (2013) on 100 billion words of Google News, and are publicly available.<sup>1</sup> We initially keep the word vectors static and learn only the other parameters of the model. Despite little tuning of hyperparameters, this simple model achieves excellent results on multiple benchmarks, suggesting that



# 1D-CNN for NLP

## ▪ Convolutional Neural Networks for Sentence Classification

- kernel size: n-gram, 참고하는 단어의 범위



# 1D-CNN for NLP

Execute 2-gram like below.(Called "Kernel size 2") -> Make one column.  
Execute 3-gram in same way.(Called "Kernel size 3") -> Make one column.  
Execute 4-gram...

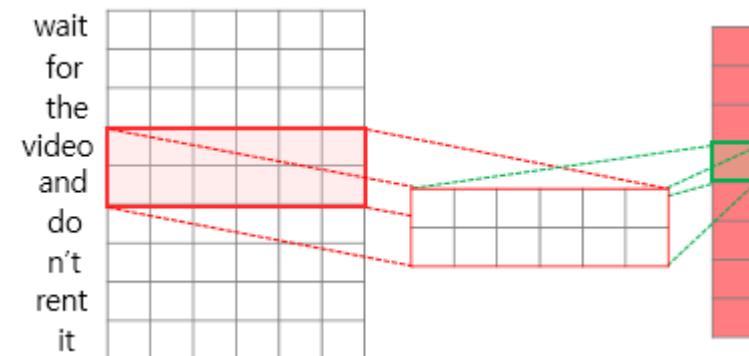
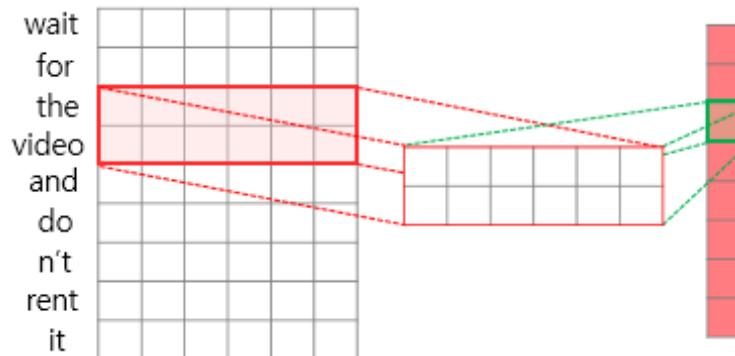
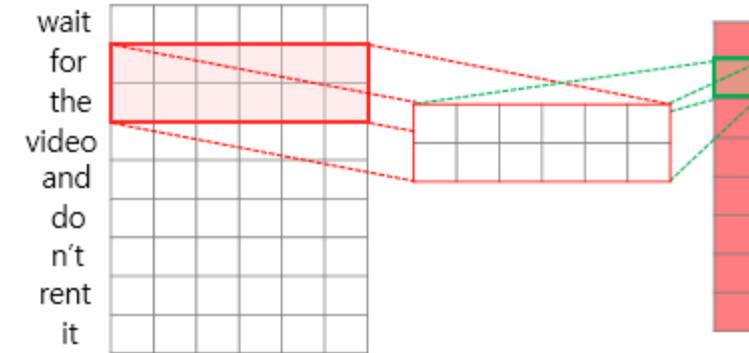
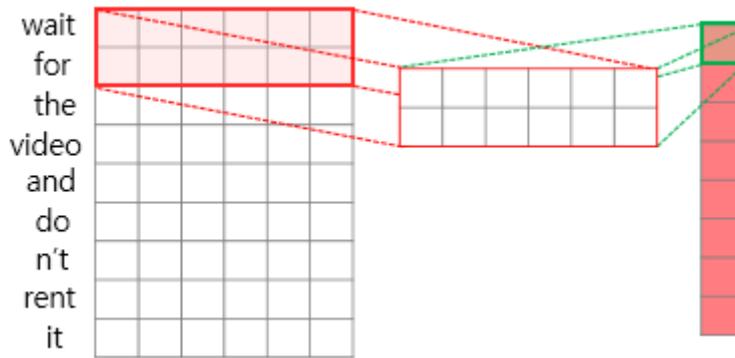
Max pooling at the results of 2-gram. -> Extract one cell.(Which has max value)  
Max pooling at the results of 3-gram.-> Extract one cell.(Which has max value)  
...

Combine them in one column.

The columns is the figure on page 207 "merged tensor"

## ▪ Convolutional Neural Networks for Sentence Classification

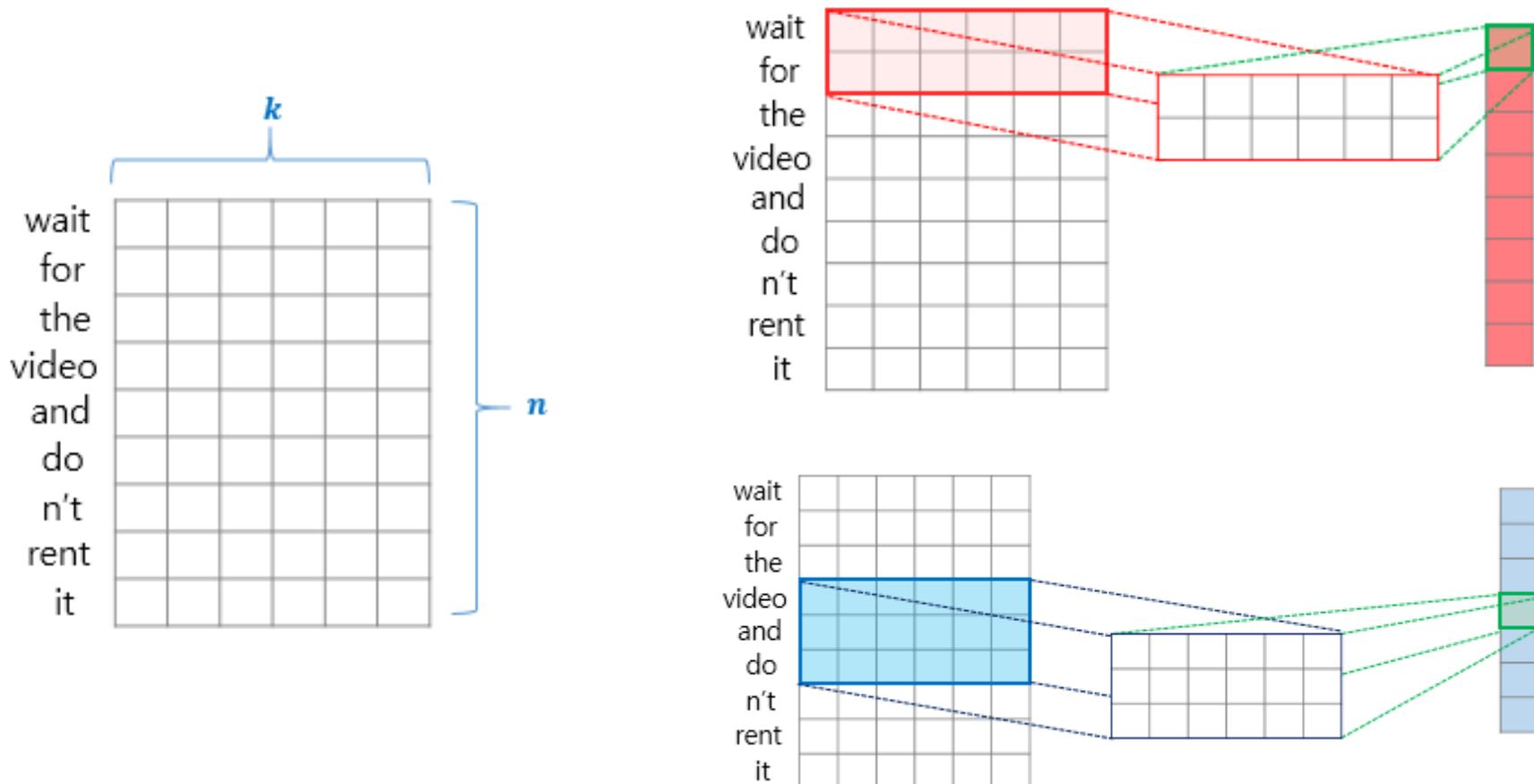
- kernel size: n-gram, 참고하는 단어의 범위



# 1D-CNN for NLP

## ▪ Convolutional Neural Networks for Sentence Classification

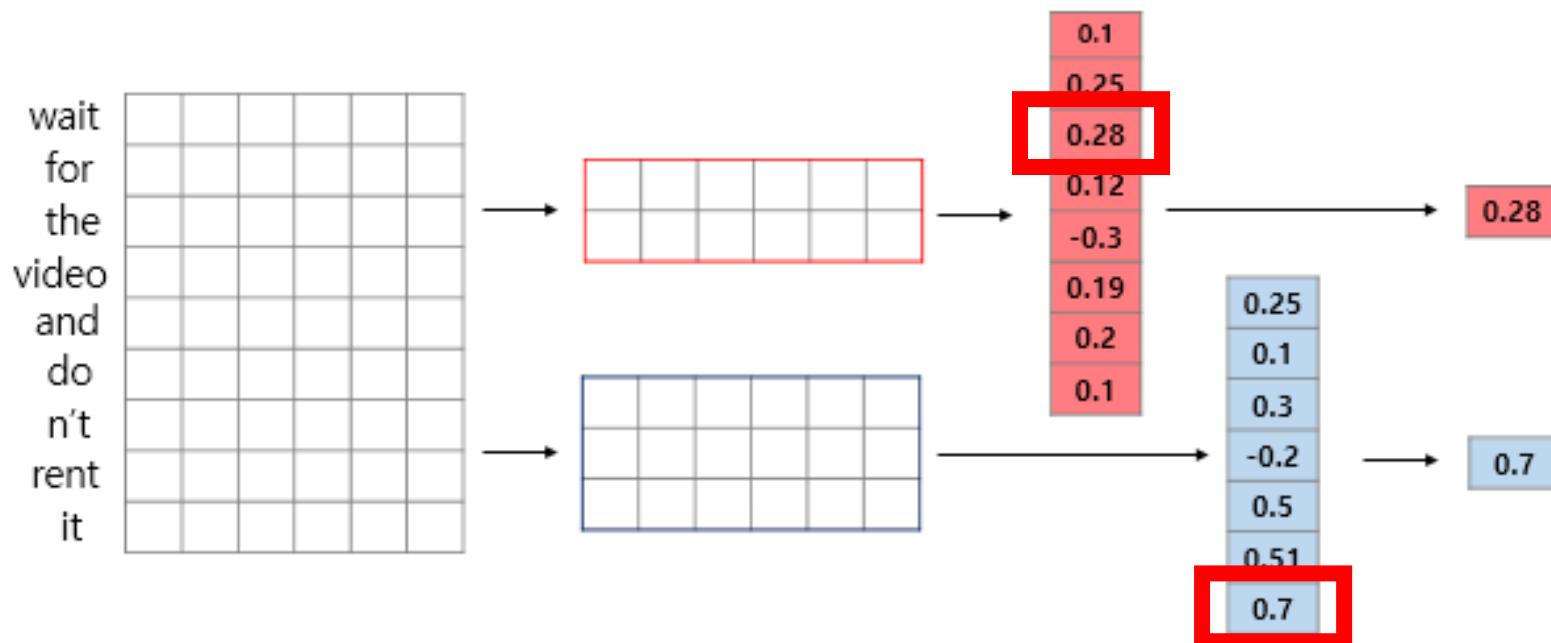
- kernel size: n-gram, 참고하는 단어의 범위



# 1D-CNN for NLP

- Convolutional Neural Networks for Sentence Classification

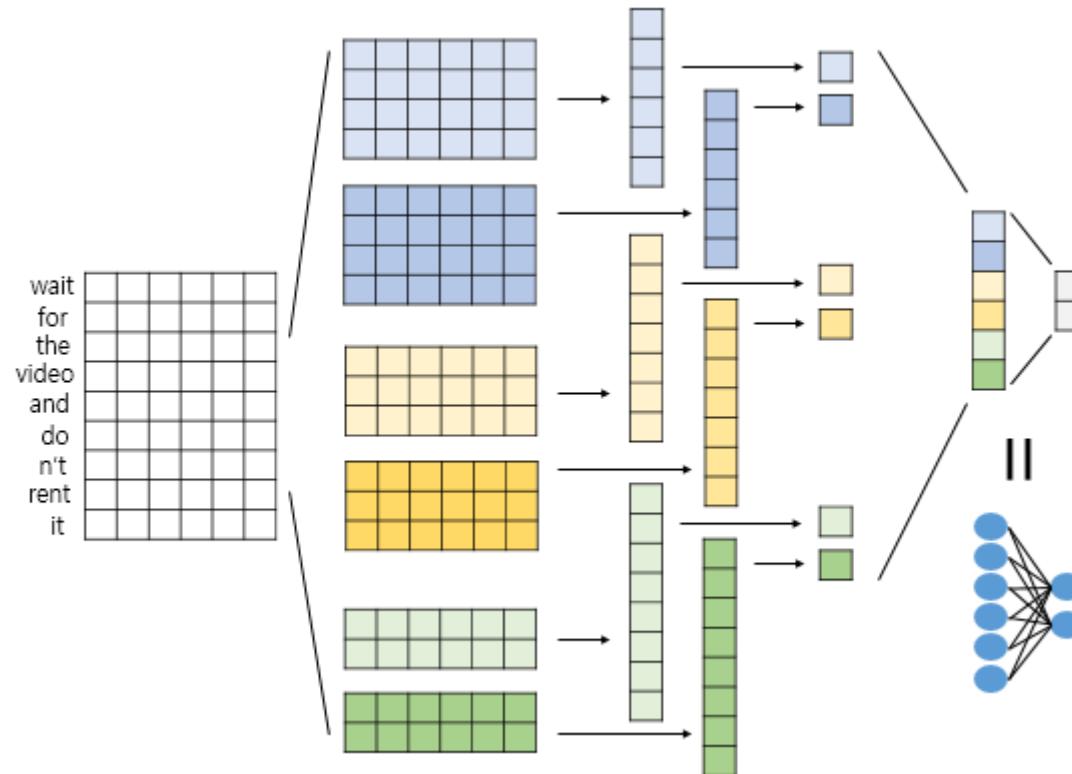
- MaxPooling



# 1D-CNN for NLP

## ▪ Convolutional Neural Networks for Sentence Classification

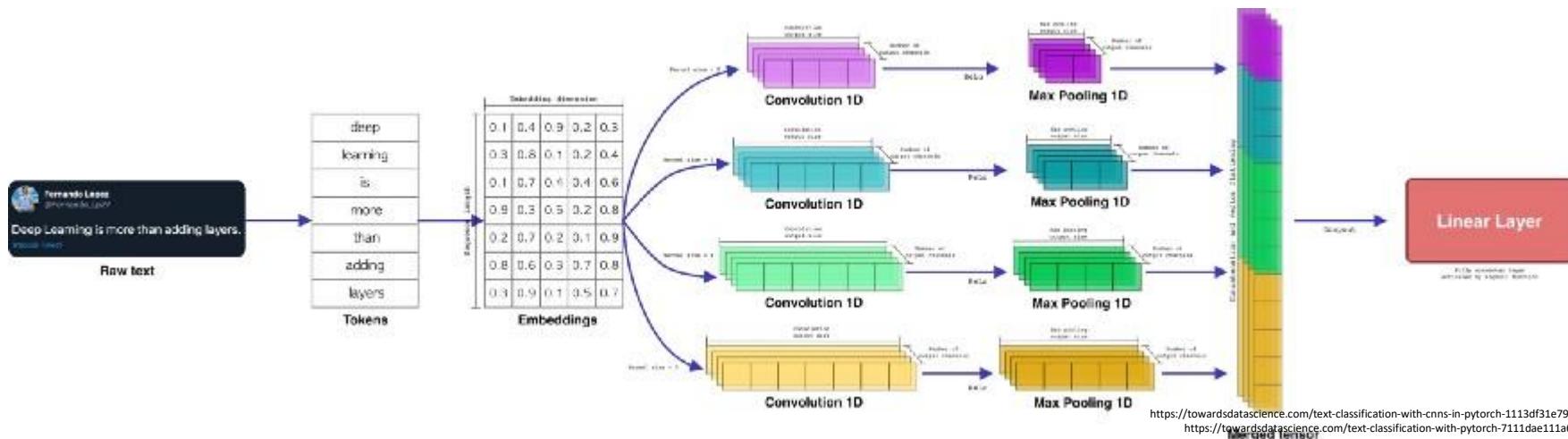
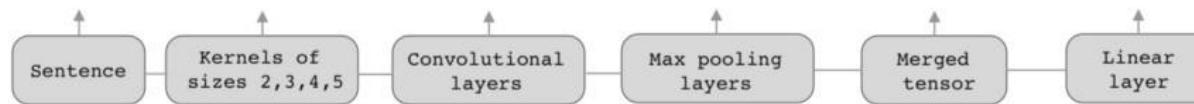
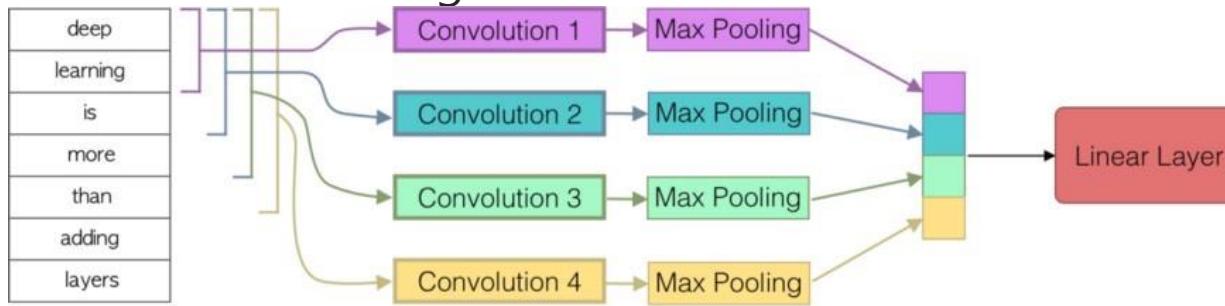
- 1D CNN + MaxPooling



# 1D-CNN for NLP

## ▪ Convolutional Neural Networks for Sentence Classification

- 1D CNN + MaxPooling



# 1D-CNN for NLP 실습

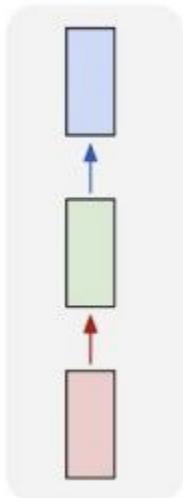
---

- **Convolutional Neural Networks for Sentence Classification**
  - `cnn_4_text_CNN_RNN_고급_colab.ipynb`

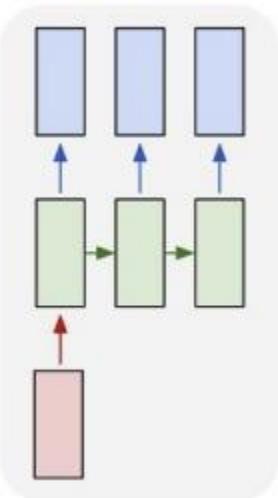
# **Recurrent Neural Network for Text Classification**

# Review of Deep Learning (RNN)

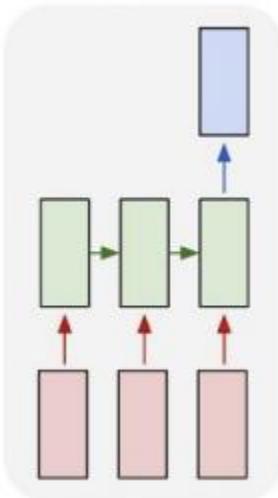
one to one



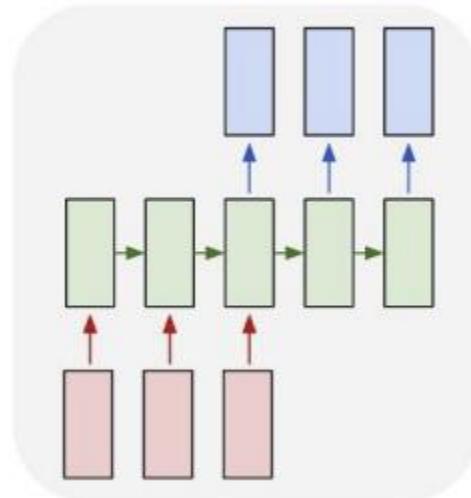
one to many



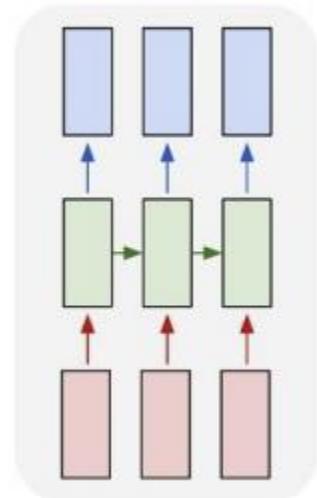
many to one



many to many



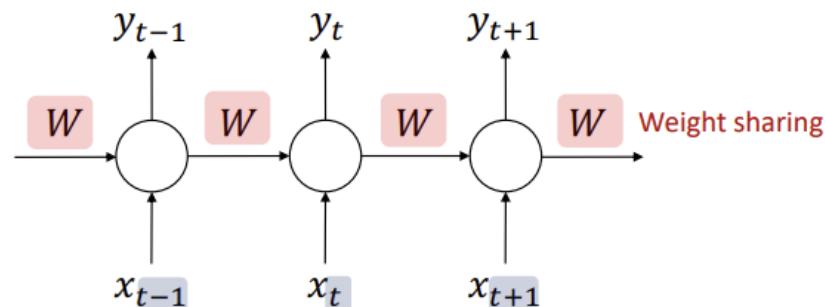
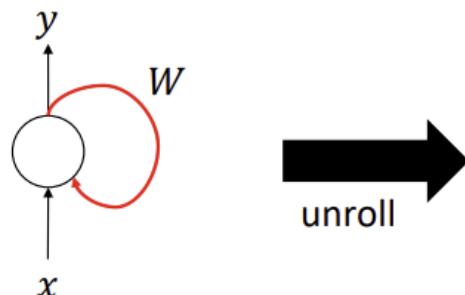
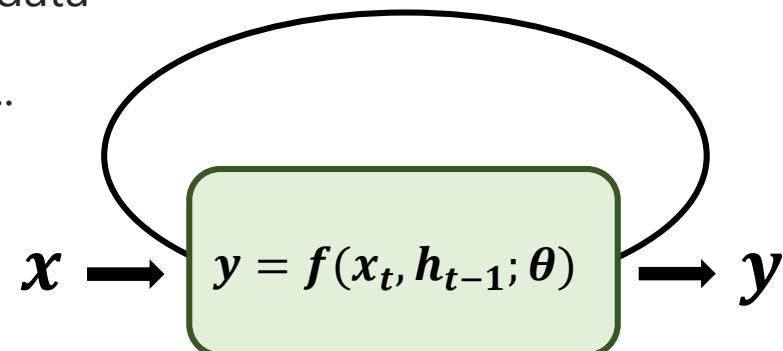
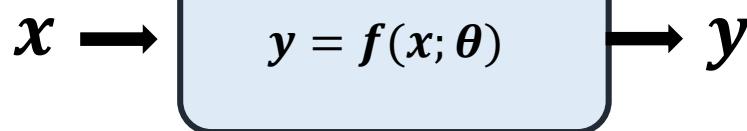
many to many



# Review of Deep Learning (RNN)

## ▪ Model type

- 기존 뉴럴넷 구조: Non sequential data
  - Image, Tabular
- Recursive한 뉴럴넷 구조: Sequential data
  - Text, sensor, stock price, sound, ...



# Review of Deep Learning (RNN)

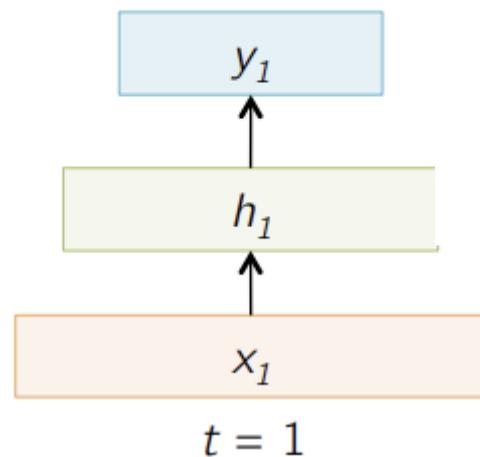
---

## ▪ Model type

- 기존 뉴럴넷 구조: Non sequential data
  - Image, Tabular
- Recursive한 뉴럴넷 구조: Sequential data
  - Text, sensor, stock price, sound, ...
- NLP는 단어들이 모여(sequence) 문장이 되고, 문장이 모여 문서
- 문장 내의 단어들은 앞뒤 위치에 따라 서로에게 영향
- 문서 내의 문장들도 위치에 따라 서로에게 영향을 주고 받음
- 시간, 순서 정보를 사용하여 입력을 학습하는 방법 : Sequence Modeling

# Review of Deep Learning (RNN)

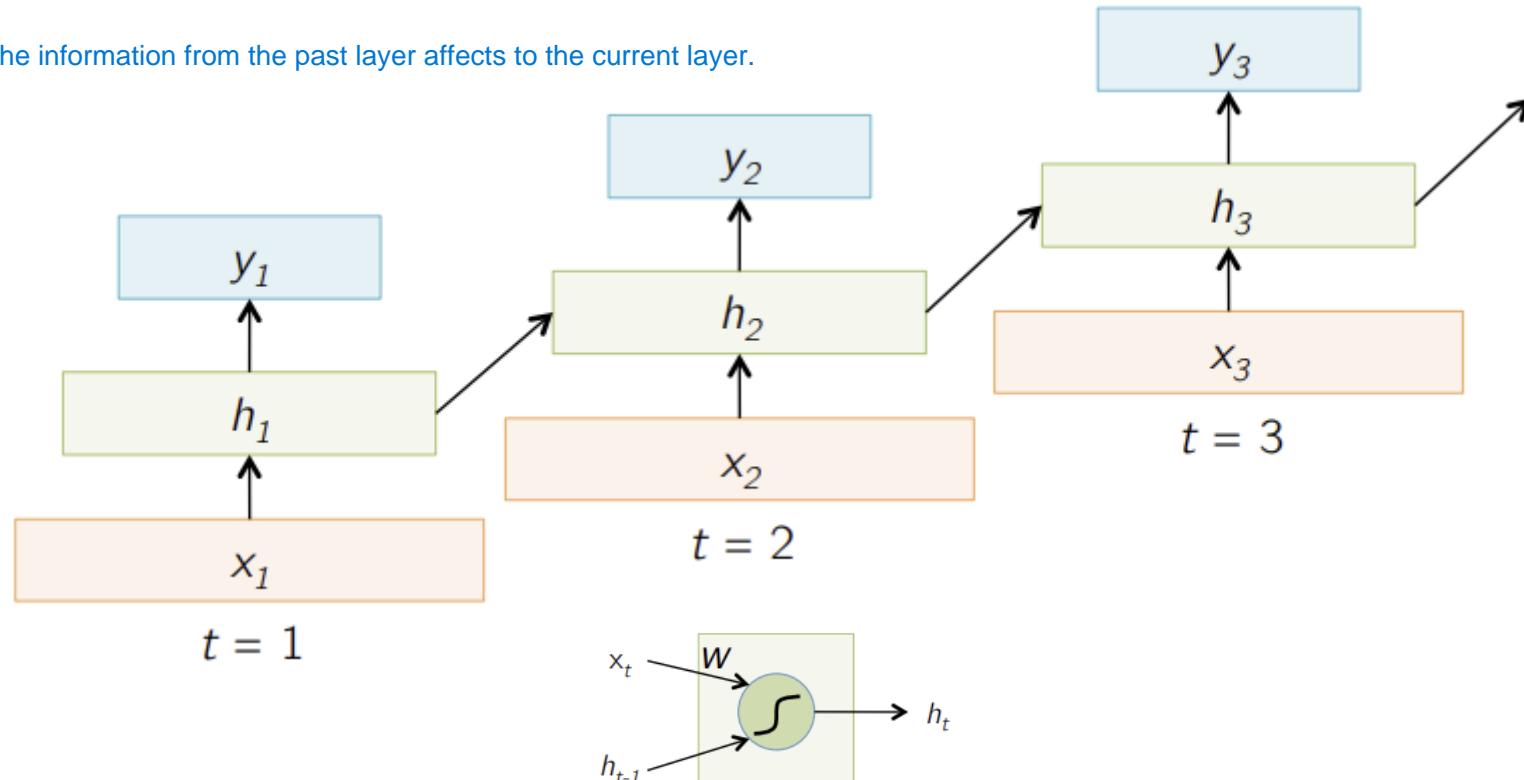
- RNN



# Review of Deep Learning (RNN)

## ■ RNN

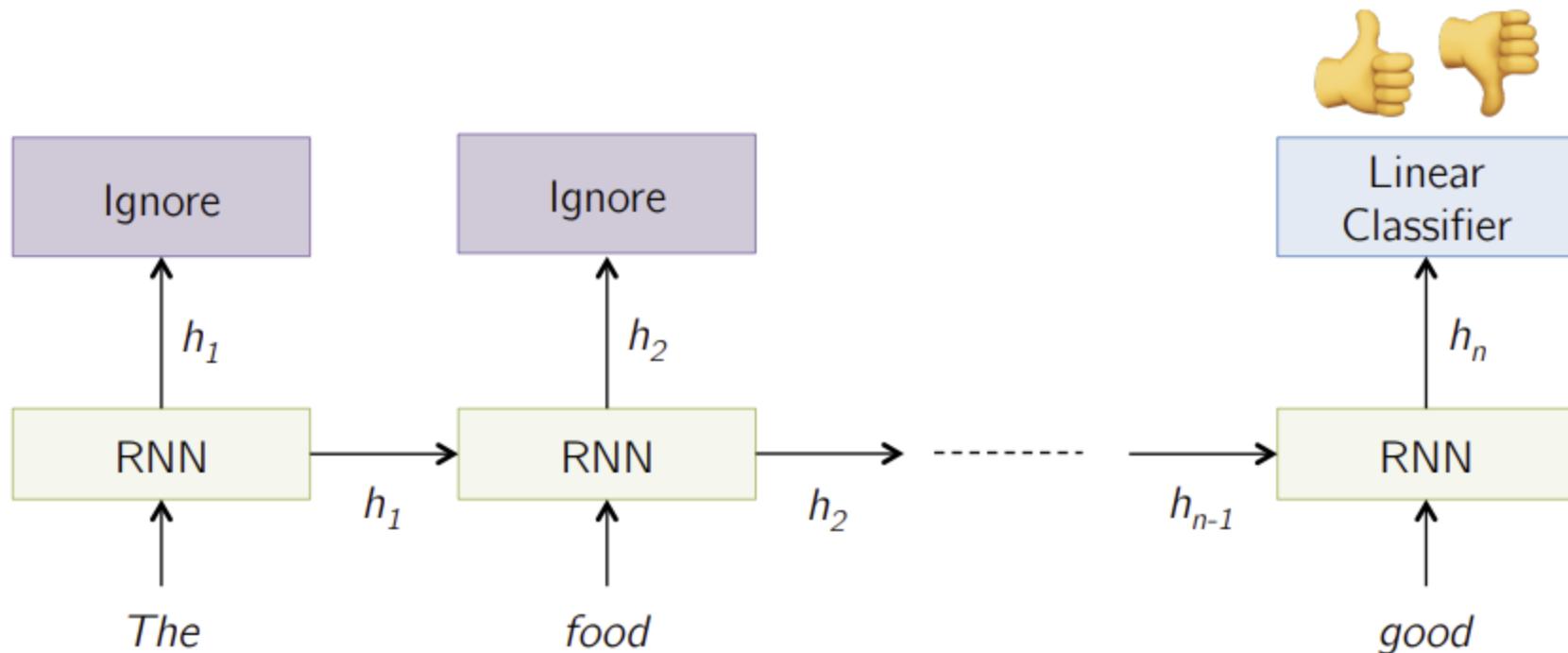
The information from the past layer affects to the current layer.



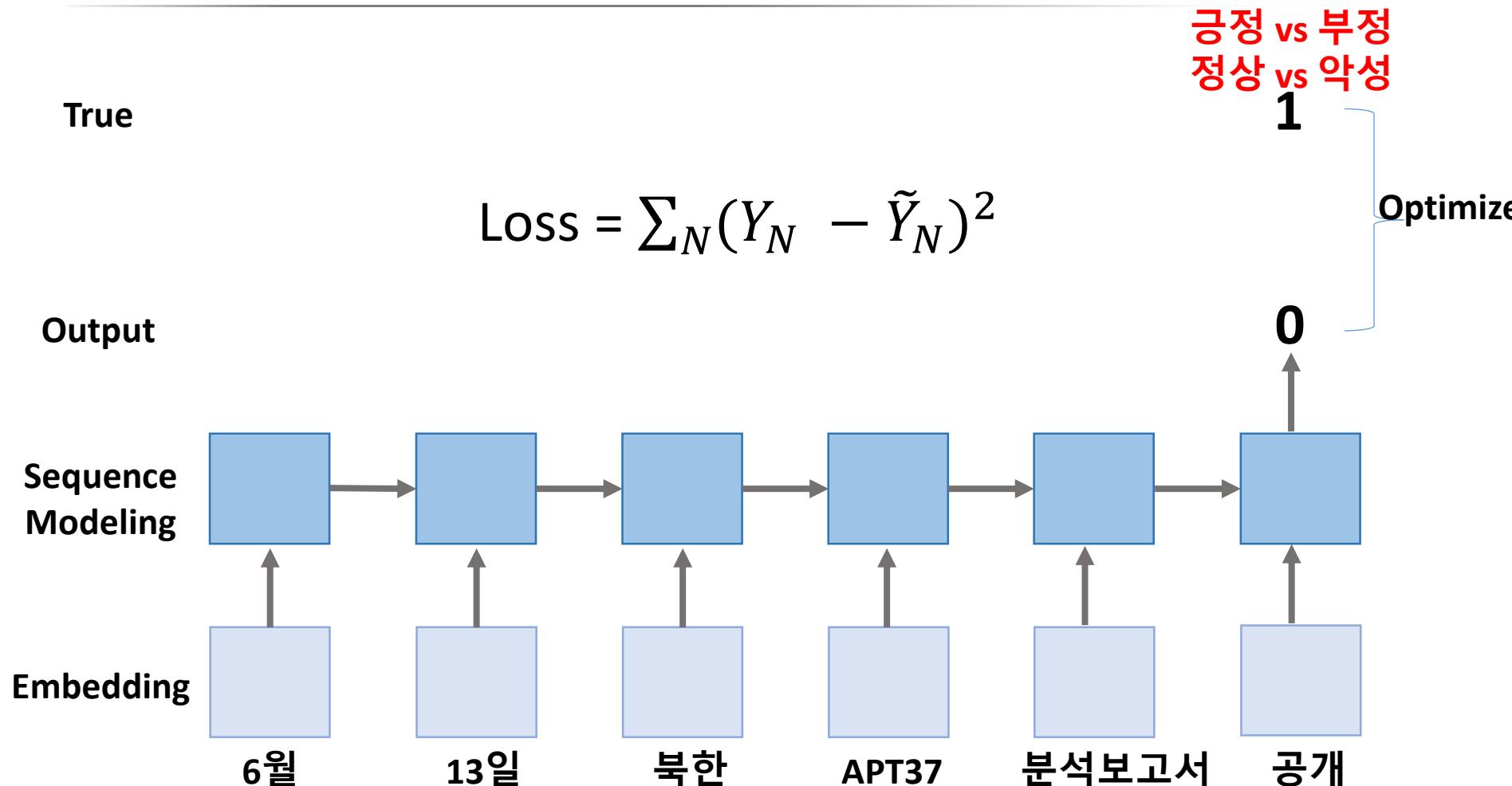
$$h_t = \tanh W \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix}$$

# Review of Deep Learning (RNN)

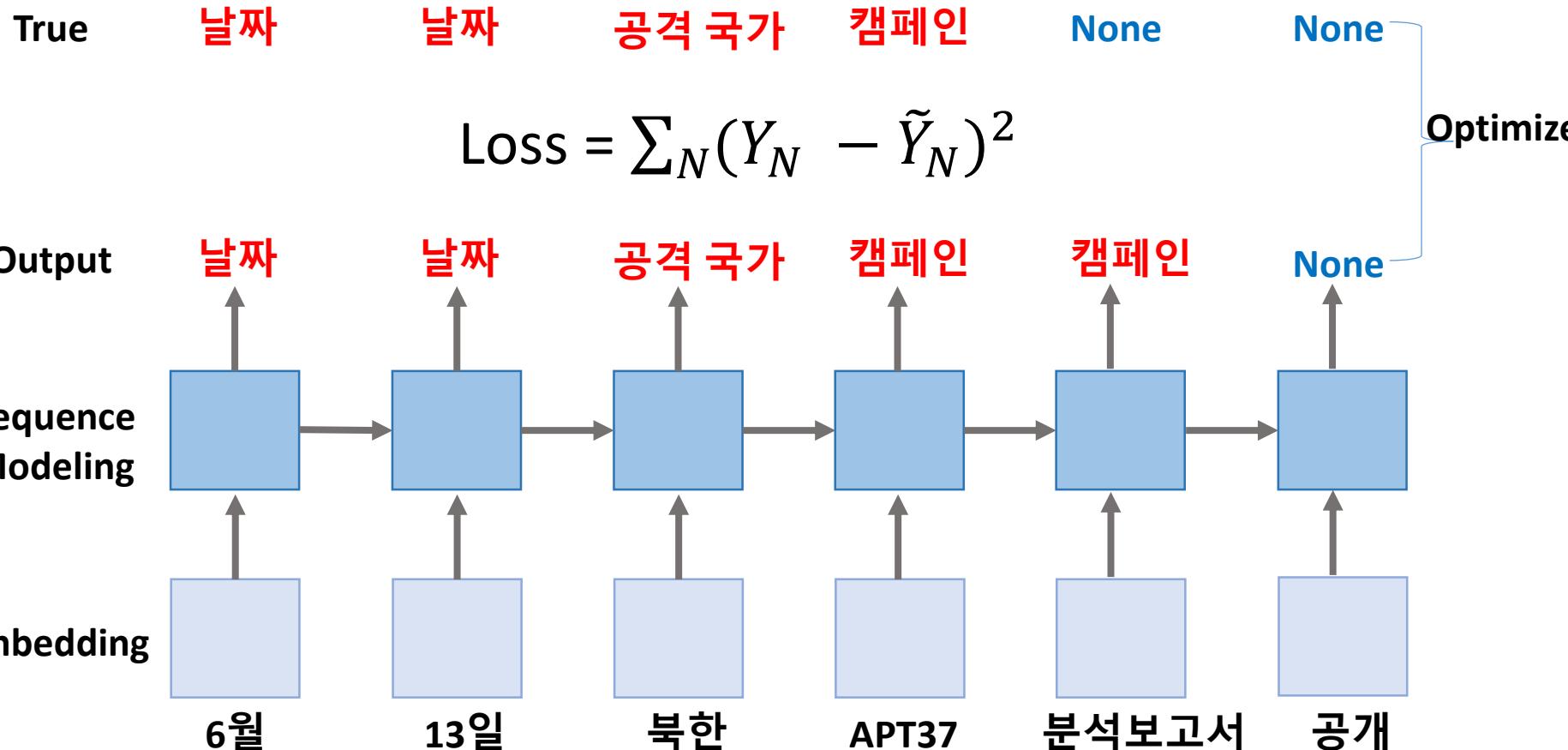
- RNN



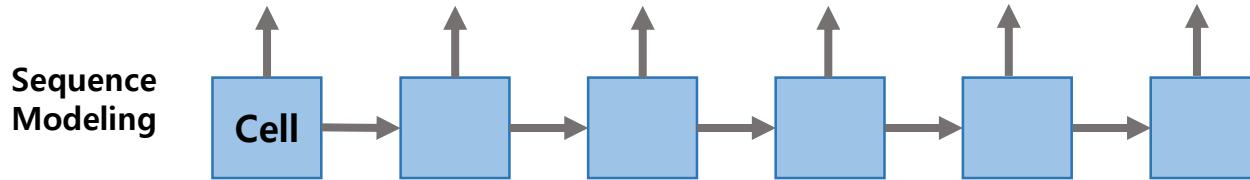
# 일반적인 NLP 문제(N to 1)



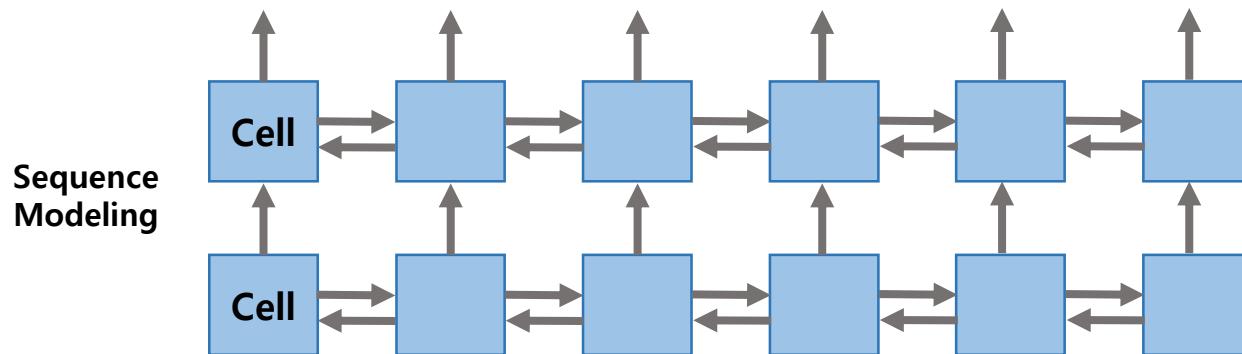
# NER기본 구조 – Sequence labeling(N to N)



# NER기본 구조 – Sequence labeling(N to N)



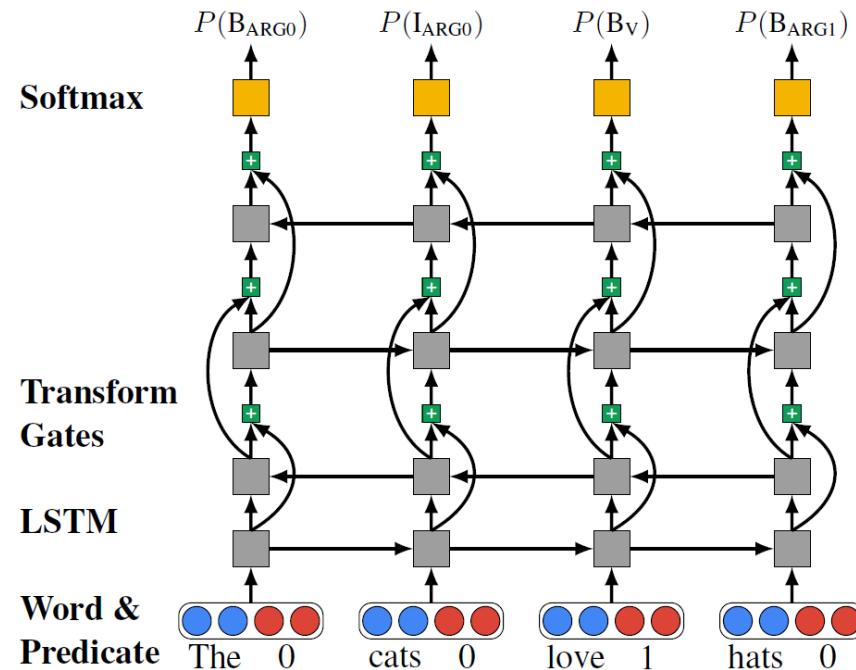
1. Cell 변경 : RNN, LSTM, GRU
2. 층 추가
3. 방향 추가: 단방향, 양방향(Bi-LSTM)



# NER기본 구조 – Sequence labeling(N to N)

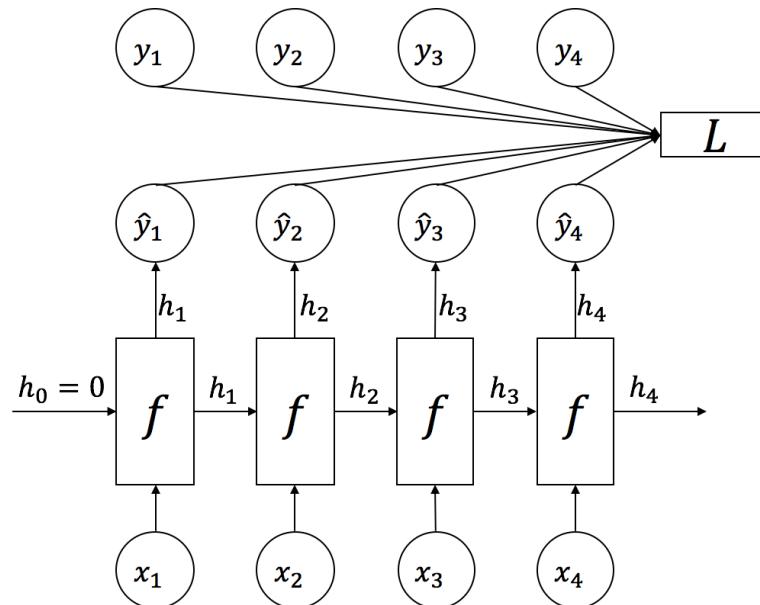
## 4. HighWay-LSTM

- He, Luheng, et al. "Deep semantic role labeling: What works and what's next." ACL. 2017.



# Recurrent Neural Network 1

- RNN

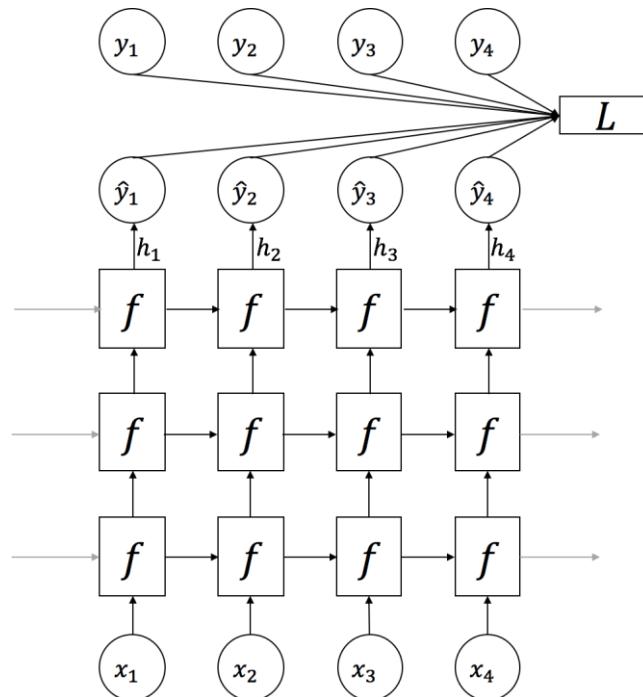


$$\begin{aligned}\hat{y}_t &= h_t = f(x_t, h_{t-1}; \theta) \\ &= \tanh(W_{ih}x_t + b_{ih} + W_{hh}h_{t-1} + b_{hh}) \\ \text{where } \theta &= [W_{ih}; b_{ih}; W_{hh}; b_{hh}].\end{aligned}$$

$$\mathcal{L} = \frac{1}{n} \sum_{t=1}^n loss(y_t, \hat{y}_t)$$

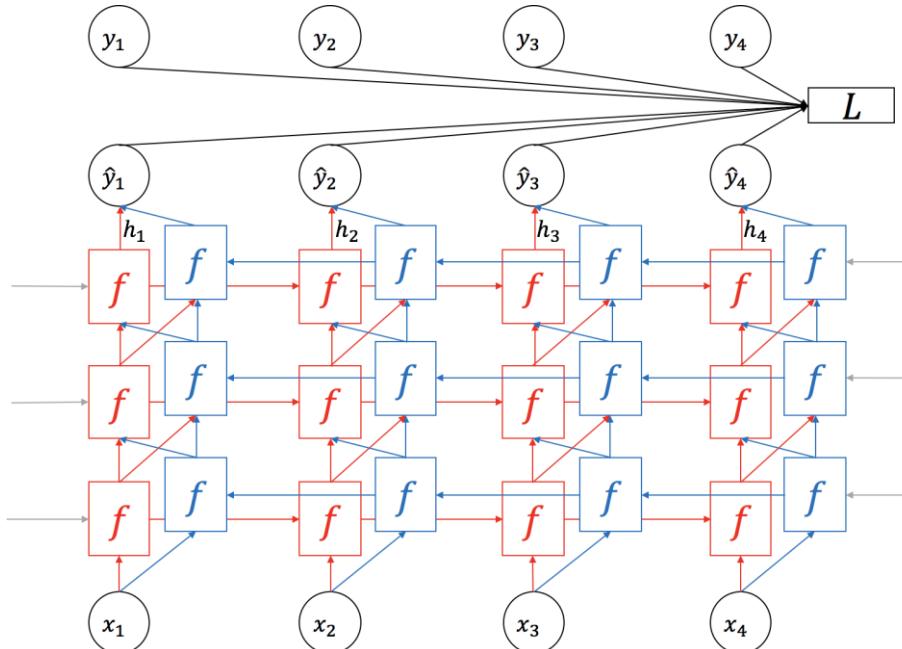
# Recurrent Neural Network 2

- Multi-layer RNN



# Recurrent Neural Network 3

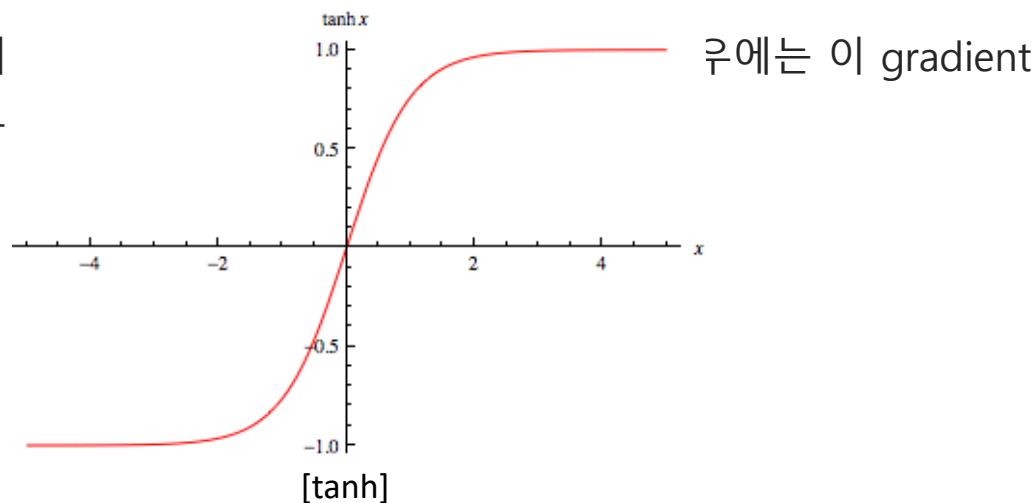
- Bi-directional RNN



# Recurrent Neural Network 4

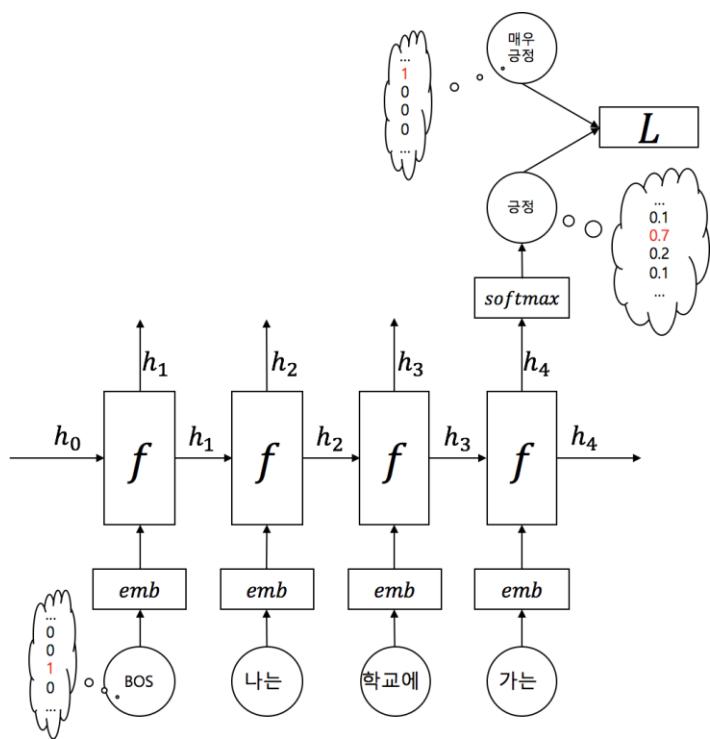
## ▪ Gradient Vanishing

- tanh, 양 끝이 수평에 가까움
  - gradient = 0
- time-step이 많거  
vanishing 문제가



# How to Apply RNN to NLP 1

- 마지막 hidden state만 출력으로 이용(N to 1)



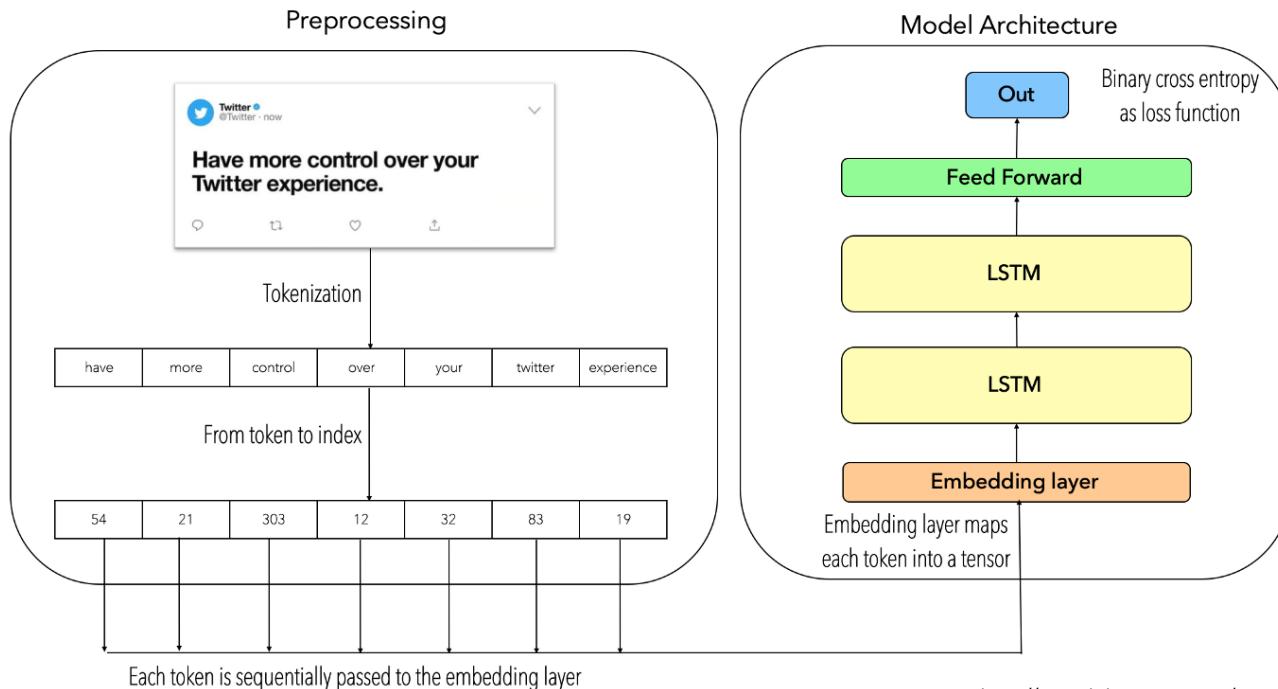
N to 1 문제가 잘 풀리면 어떤분야에 적용할 수 있나?

언어모델	대한	민국	→	만세	
대화모델	알람	좀	→	Set.Alarm	
감성분석	이	영화	→	Positive	
문서분류	W	W	...	→	Topic=music
Zero Pronoun	W	W	...	→	Pronoun Dropped

# Review of Deep Learning (RNN)

## ■ RNN

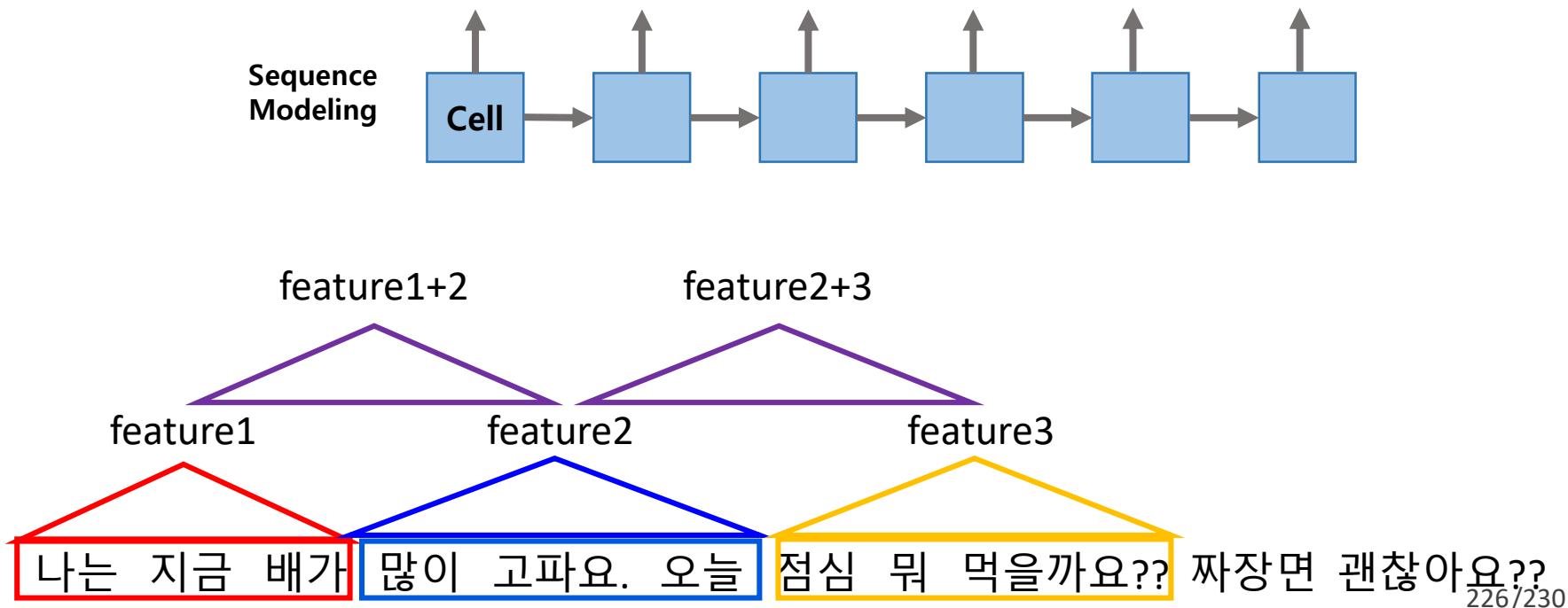
- rnn\_1\_text\_RNN\_colab.ipynb
- cnn\_4\_text\_CNN\_RNN\_고급\_colab.ipynb



# Review of Deep Learning (CNN/RNN)

- DL는 입력 텍스트를 embedding하여 vector화 함

- CNN: 주변 단어(kernel size)를 종합적으로 판단해서 임베딩
- RNN: 단어의 순서 정보로 임베딩

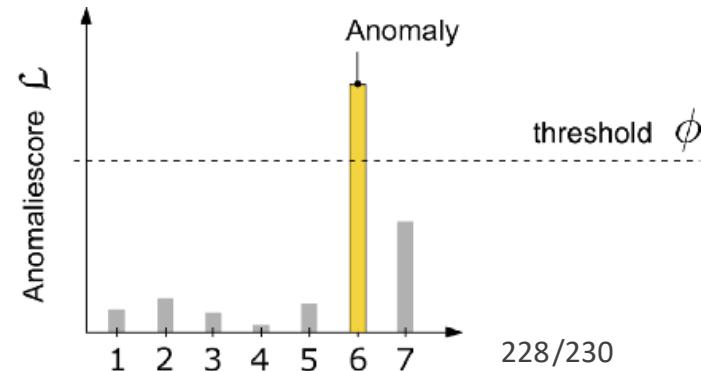
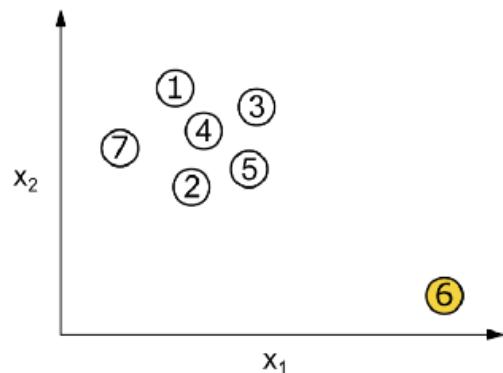
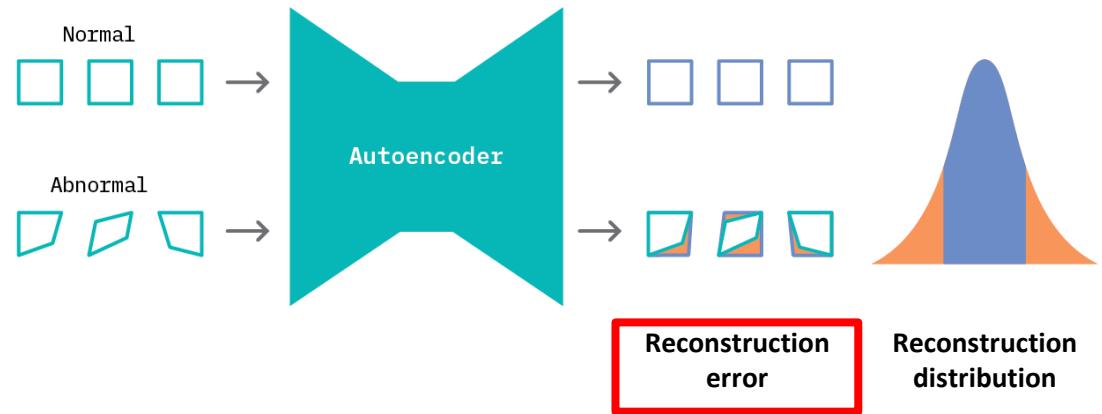


# AutoEncoder

Encoder-Decoder model

# AutoEncoder

- Encoder
- Decoder
- Encoder-Decoder



# AutoEncoder 실습

---

- colab 실습
- URL: [https://uvadlc-notebooks.readthedocs.io/en/latest/tutorial\\_notebooks/tutorial9/AE\\_CIFAR10.html](https://uvadlc-notebooks.readthedocs.io/en/latest/tutorial_notebooks/tutorial9/AE_CIFAR10.html)

---

감사합니다