

인하대학교 텍스트 데이터를 통해 알아본

---

# 자연어처리(NLP)란 무엇인가

---

팀번호 : 16

12181912 김현수  
12191909 장재현

# 목차

## Table of contents

- NLP(Natural Language Processing)
- Language Model (언어모델)
- NLP의 활용 (감성분석 예시)
- 코드 설명
- 아쉬운점 및 개선사항
- NLP와 통계학

# 목차

## Table of contents

### ■ NLP(Natural Language Processing)

#### ■ Language Model (언어모델)

#### ■ NLP의 활용 (감성분석 예제)

#### ■ 코드 실행

#### ■ 아쉬운점 및 개선사항

#### ■ NLP와 통계학

# 통계학 (Statistics)

산술적 방법을 기초로 하여, 주로 다량의 데이터를 관찰하고 정리 및 분석하는 방법을 연구하는 수학의 한 분야 (위키백과)

## Data(데이터)?

# Data(데이터)



속성  연속형 데이터  
 이산형 데이터

A	B	C	D	E	F	G	H
1	Category	Mathematics	Math	Area	Area	Area	Area
2	All Scores	Lowest co-Silver	189	1206	297	5404	291
3	All Scores	Lowest co-Silver	186	4581	251	6208	291
4	All Scores	Lowest co-Silver	195	7506	319	8435	369
5	All Scores	Lowest co-Silver	245	7813	387	6573	387
6	All Scores	Lowest co-Silver	242	6912	387	1509	387
7	All Scores	Lowest co-Silver	156	8932	248	5531	248
8	All Scores	Lowest co-Silver	146	2335	205	6254	205
9	All Scores	Lowest co-Silver	177	5689	289	266	289
10	All Scores	Lowest co-Silver	186	3349	285	1556	285
11	All Scores	Lowest co-Silver	160	82	237	6117	237
12	All Scores	Lowest co-Silver	132	1475	200	1864	200
13	All Scores	Lowest co-Silver	145	9009	211	529	211
14	All Scores	Lowest co-Silver	143	9383	234	4514	234
15	All Scores	Lowest co-Silver	177	4174	278	3976	278
16	All Scores	Lowest co-Silver	193	8379	319	5371	319
17	All Scores	Lowest co-Silver	226	5843	361	4241	361
18	All Scores	Lowest co-Silver	152	0919	239	6149	239
19	All Scores	Lowest co-Silver	153	2844	241	7323	241
20	All Scores	Lowest co-Silver	176	612	245	7512	245
21	All Scores	Lowest co-Silver	171	3506	269	8435	269

대상  정형 데이터 : 미리 정해진 구조에 따라 저장된 데이터 (Structured Data)  
 비정형 데이터 : 정해진 구조가 없이 저장된 데이터 (Unstructured Data)

척도  질적 데이터  
 양적 데이터



이미지 데이터  
(Computer Vision)



비정형 데이터  
(Deep Learning)

텍스트 데이터  
(Natural Language Processing)



비정형 데이터  
(Deep Learning)



이미지 데이터  
(Computer Vision)



텍스트 데이터  
(Natural Language Processing)

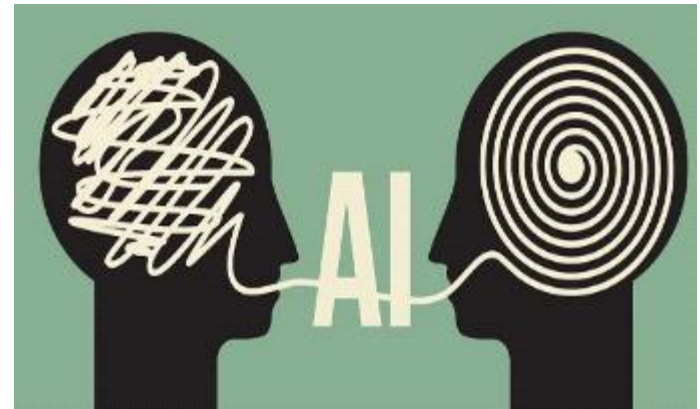


# 자연어처리 (Natural Language Processing)

## 자연어처리란?

인간의 언어 현상을 컴퓨터와 같은 기계를 이용해서 묘사할 수 있도록 연구하고 이를 구현하는 인공지능의 주요 분야 중 하나 (위키백과)

- 기계 번역 (Machine Translation)
- 내용 요약 (~~Text Summarization~~)
- 감성 분석 (~~Sentiment Anlaysis~~)
- 질의 응답 (~~Question Answering~~)
- 텍스트 분류 (Text Classification)
- 챗봇 시스템 (Chat-Bot System)
- 음성 인식 (Speech Recognition)





# 목차

## Table of contents



NLP(Natural Language Processing)



Language Model (언어모델)



NLP의 활용 (감성분석 예시)



코드 실행



어휘분석 및 계산서형



NLP와 통계학

# Language Model (언어모델)

## Language Model이란?

: 단어 시퀀스에 대한 확률분포(probability distribution)

말뭉치 (Corpus)  
(746530개의 단어 사전)

$P(\text{인하대학교는 대한민국 사립 종합대학이다}) = 0.0001$

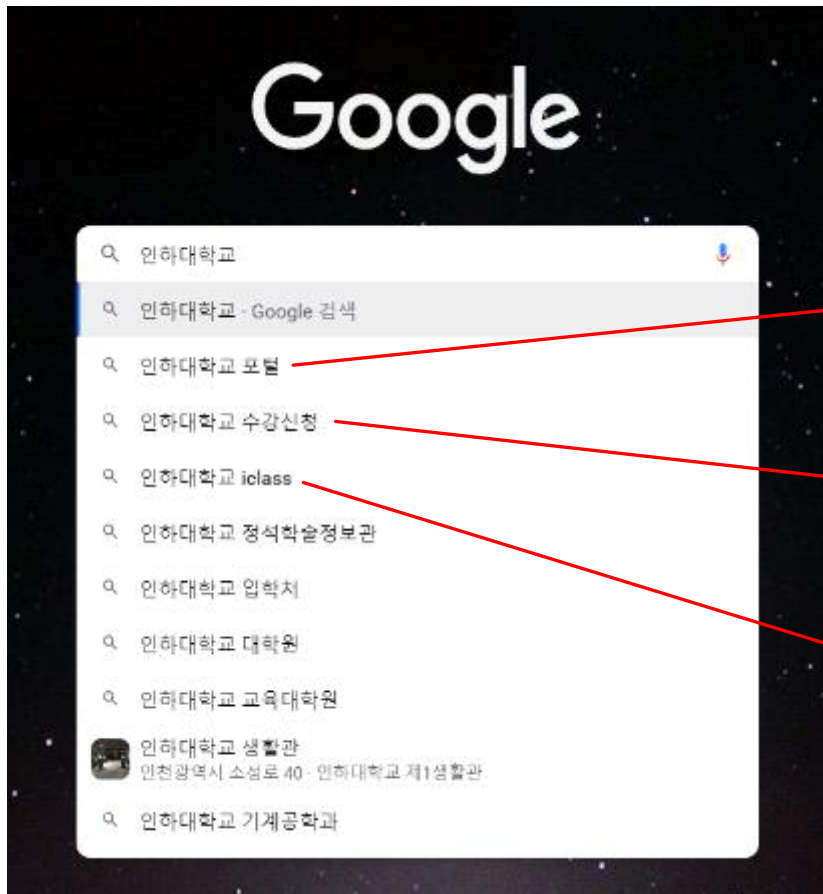
$P(\text{대한민국 사립 종합대학은 인하대학교다}) = 0.00000001$



# 언어 모델의 활용

조건부확률의 연쇄법칙 (Chain rule for conditional probability)

$$P(x_1, x_2, x_3 \dots x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2) \dots P(x_n|x_1 \dots x_{n-1})$$



$$P(\text{인하대학교, 포털}) = P(\text{인하대학교}) * P(\text{포털} | \text{인하대학교})$$

$$P(\text{인하대학교, 수강신청}) = P(\text{인하대학교}) * P(\text{수강신청} | \text{인하대학교})$$

$$P(\text{인하대학교, iclass}) = P(\text{인하대학교}) * P(\text{iclass} | \text{인하대학교})$$

(검색 엔진에서의 검색어 자동완성 기능)

# 언어 모델의 활용

## 1. 기계 번역

$P(\text{나는 버스를 탔다}) > P(\text{나는 버스를 태운다})$

: 언어 모델은 두 문장을 비교하여 좌측의 문장의 확률이 더 높다고 판단합니다.

## 2. 오타 교정

$P(\text{내 동생은 놀이터로 달려갔다}) > P(\text{내 동생은 놀이터로 깔려갔다})$

: 언어 모델은 두 문장을 비교하여 좌측의 문장의 확률이 더 높다고 판단합니다.

## 3. 음성 인식

$P(\text{나는 메론을 먹는다}) < P(\text{나는 메론을 먹는다})$

: 언어 모델은 두 문장을 비교하여 우측의 문장의 확률이 더 높다고 판단합니다.

## 데이터 수집 (인하대 데이터)

(인하소식) 인하뉴스 2591개

(인하소식) 공지사항 2294개

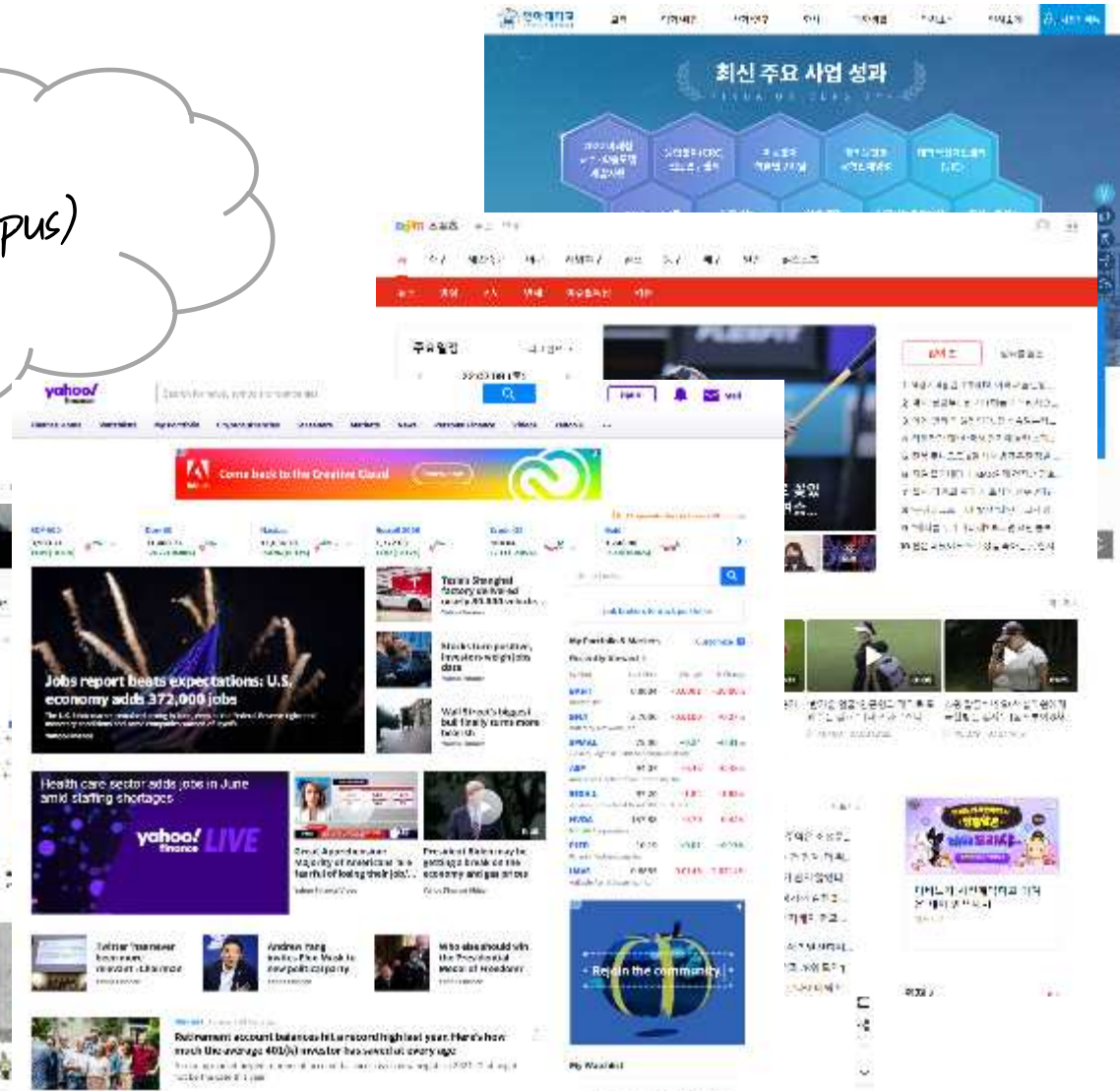
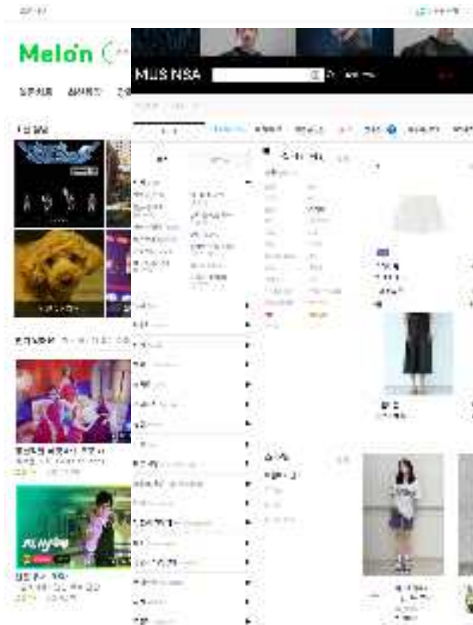
“(인하광장) 자유게시판 14156개”

**총 19041개의 게시물**

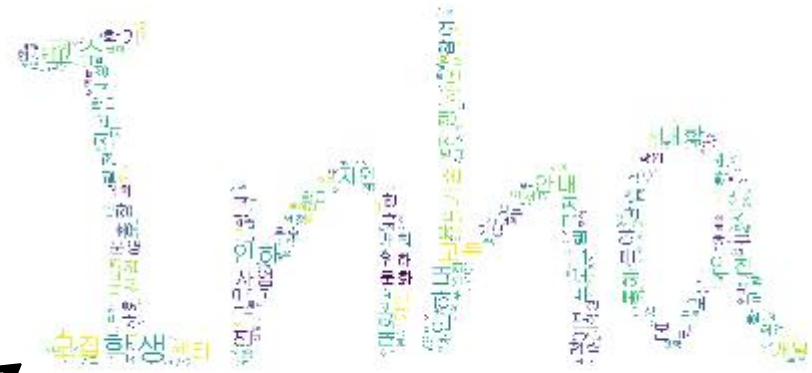
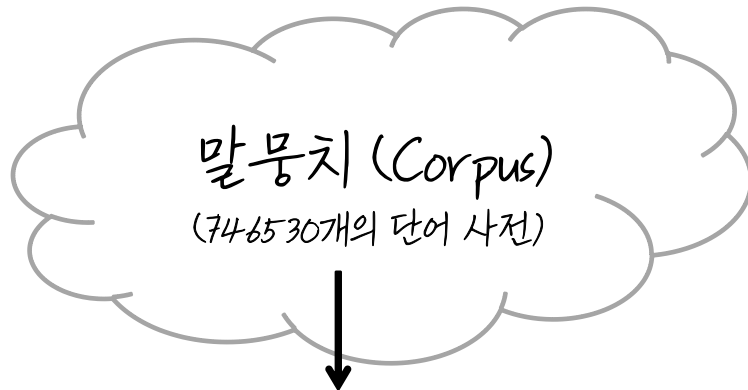
(746530개의 단어)

# Language Model (언어모델)

말뭉치 (Corpus)  
( ... ??? )



# Language Model (언어모델)



(워드클라우드 예시)

'학생' 의 등장 확률은  $0.0122$  ( $9117/746530$ )

'교수' 의 등장 확률은  $0.0107$  ( $8015/746530$ )

~~'등' 의 등장 확률은  $0.0079$  ( $5878/746530$ )~~

'모집' 의 등장 확률은  $0.007$  ( $5238/746530$ )

'인하' 의 등장 확률은  $0.0068$  ( $5085/746530$ )

'교육' 의 등장 확률은  $0.0065$  ( $4889/746530$ )

'연구' 의 등장 확률은  $0.0061$  ( $4573/746530$ )

'인하대' 의 등장 확률은  $0.0061$  ( $4557/746530$ )

'인천' 의 등장 확률은  $0.0061$  ( $4531/746530$ )

'대학' 의 등장 확률은  $0.0059$  ( $4404/746530$ )

→ 불용어(Stopword)처리

" 조사, 접미사 같은 단어들은 문장에서는 자주 등장하지만  
실제 의미 분석을 하는데는 거의 기여하는 바가 없는 단어 "

ex) 것, 그, 등, 있, 들, 내 ...

# 목차

## Table of contents



NLP(Natural Language Processing)



Language Model (언어모델)



NLP의 활용 (감성분석 예시)



코드 설명



주요문법 및 계산사항



NLP와 통계학



# 감성 분석 (Sentiment Analysis)

감성 분석 (Sentiment Analysis)이란 텍스트에 드러나는 의견이나 감성 표현 정도 등의 주관적인

[감성 분석 데이터 (

id document  
83759700 00 나뭇...  
8819312 흥 포스터  
10035343 너덕  
9045010 고드름 이야  
6460659 사-O 무비 그  
5403914 그 끝은 비  
7737314 온 초열 건강  
6443047 온 나라도 O  
7159791 리슨이 없  
5912145 리케 평평이  
8008700 온두 피니트  
110717528 볼  
5857425 온종일 손  
6619327 음력하고 양  
6954037 주장은 온  
6852435 농 다변  
8143168 온 사람들  
4831478 온이 러닝  
7435433 온 정만  
3993140 리온자는 위  
4581211 그들 식오  
7713846 온종일 도시  
8705777 저기 있다 지  
471131 온대 평일  
6490260 주제는 온

번호	종류	제목	작성자	작성일	댓글	리플	조회수
세	물집	[인공지능융합연구센터] 고성능 GPT 세팅 신청 안내 (상세 접수)	한미연	2022.06.29	0	0	121
세	모집	[2022유망기업홍보캠페인] 2022학년도 2학기 100명정기인원실습 참여확보 모집 안내	이다정	2022.05.10	4	1	400
13901	모집	인하대학교 교수학습개발센터 전문연구원 채용	민준터	2022.07.26	0	0	17
13900	물집	[두류 새강 이원디] 인제안 비전공자를 위한 동서대 및 집지대 인동거 강의	민지은	2022.07.26	1	0	38
13899	보지	[과열연세 신변가동지원인력 교육사업(신남TLO주최)] 참여 희망수요 설문조사(2022년 3차 모집기간:~2022.06.06.)	이소진	2022.07.25	0	0	34
13898	모집	[인하대보통] 건강인 사업대상자를 모집합니다.	이형진	2022.07.25	0	1	67
13897	모집	[조봉 수학자료] 조봉 수학자료 신청금 구입안내	고보경	2022.07.24	0	0	108
13896	물집	[사범대학 감사노무위원회] 2022-1학기 사범대학 정기감사 결과 보고	김종영	2022.07.24	1	0	62
13895	물집	[아이디이벤트]도전상자있는 학생과 아이디이벤트 참여인원을 가져옵니다.	이준하	2022.07.23	15	1	181
13894	모집	[산업통상자원부] 스마트디지털생태계(이민정관 부관)내 학생을 모집	주호인	2022.07.22	0	0	15
13893	물집	[물문주사] 인하대직신문 1300이 가난 '인하대직신문은 없다' 물문주사 <수신을 통해 신청 중> - 08. 01	이세준	2022.07.22	4	0	50
13892	모집	[인하대직신문] 미디어콘텐츠가치로 모집합니다. (모집기간: 2022.07.22~2022.07.29)	이재준	2022.07.22	2	0	44
13891	외역	[올해의 유망기업] 위해 신청서 접수 안내	김진원	2022.07.22	4	0	140
13890	보지	[미래시류리더학년] 2022 학제제식 선거도 skill up 데스도 에누마로 이용한 시공수행 논문 <출발 미래>	이준희	2022.07.21	1	0	60
13889	모집	[미래차융합사업단] 제2회 혁신공유대학 제2회 공모전	이준희	2022.07.21	1	0	32
13888	물집	[학생TF] 7차 가짜 세팅 가이드라인 안내	김민준	2022.07.21	4	1	294
13887	물집	[남한대학교] 7월 21일 휴강 안내	비비드	2022.07.21	0	2	94

<< 1 2 3 4 5 6 7 8 9 10 >>

작성 삭제

100

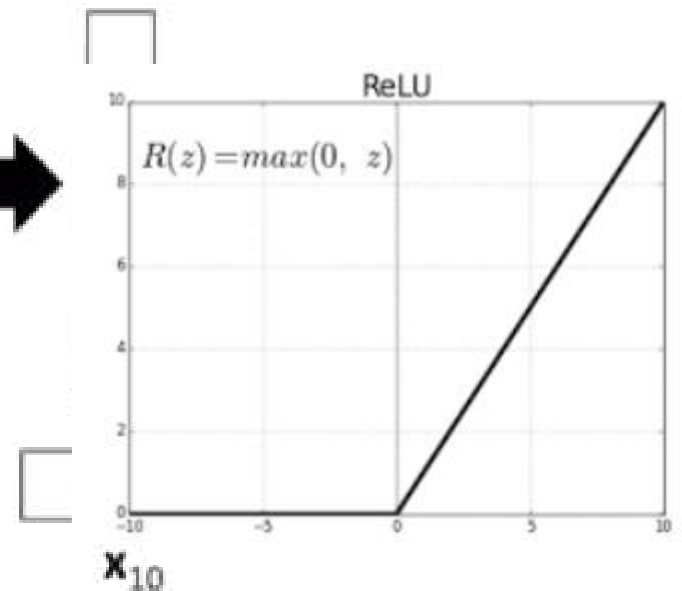
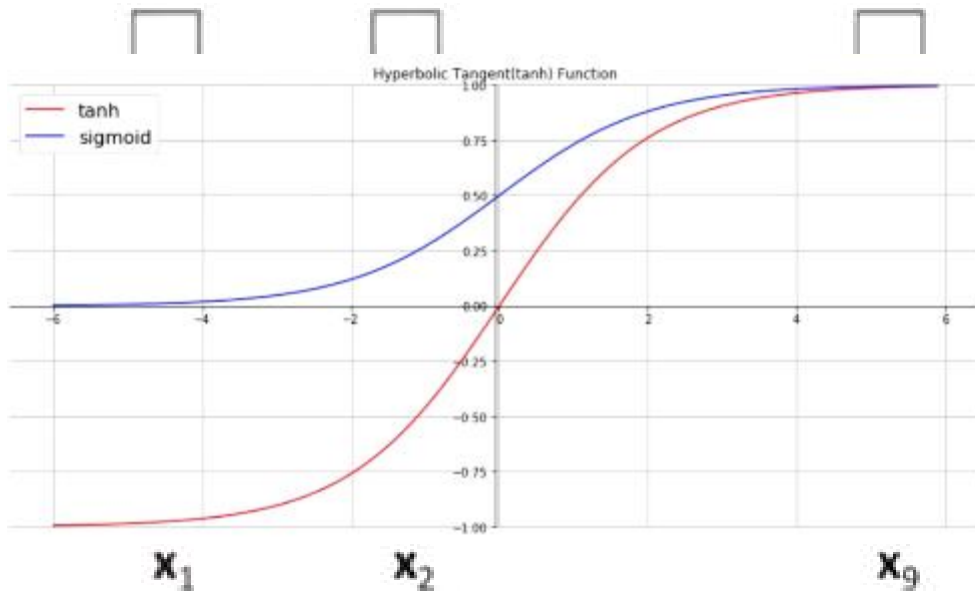
모든새마도 있는

# 모델 설명 - GRU

Example

"This movie is as impressive as a preschool Christmas play" → 긍정?(1) 부정?(0)

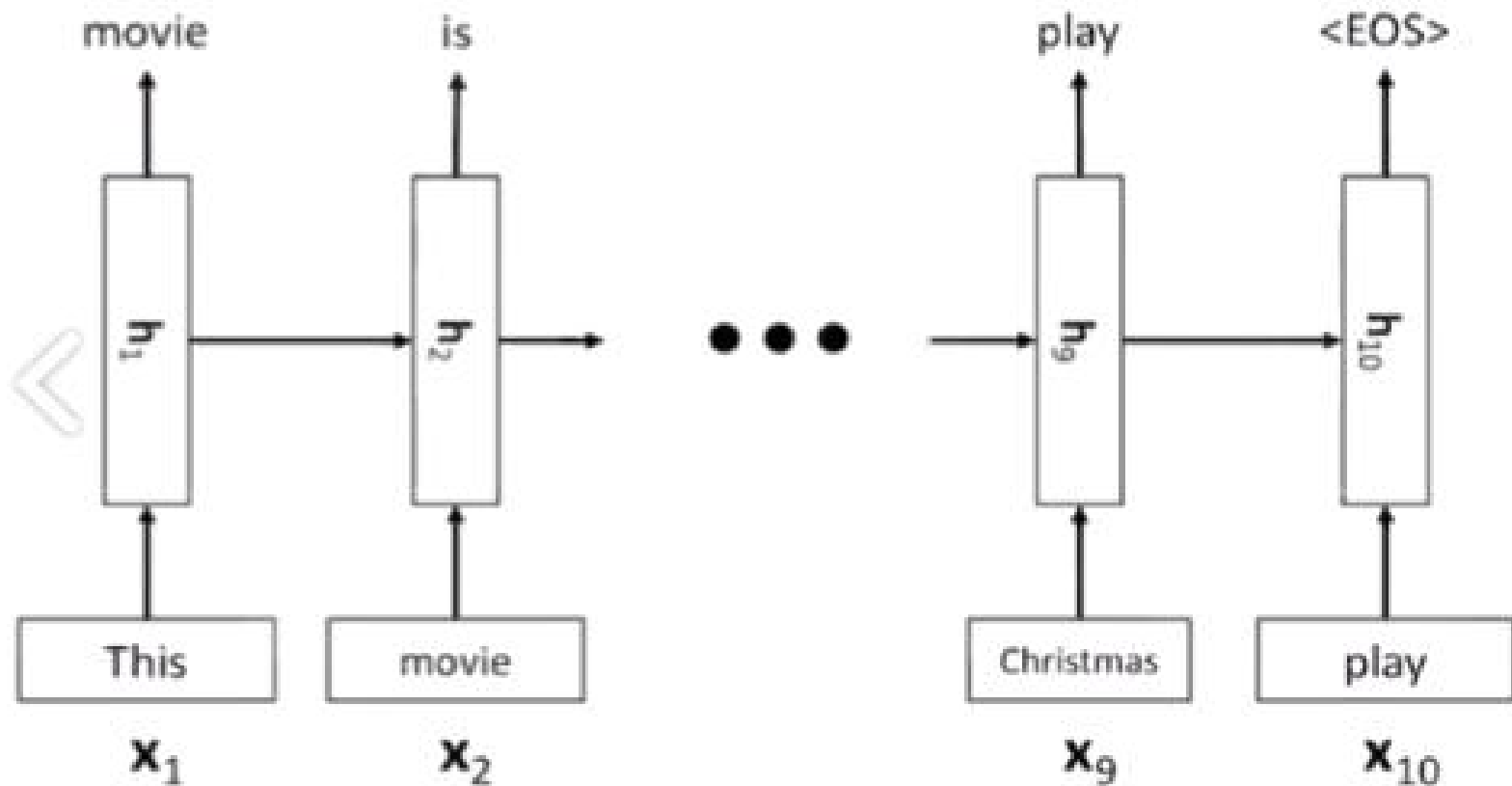
RNN (Recurrent Neural Network) : 순환신경망



# 모델 설명 - GRU

## Example

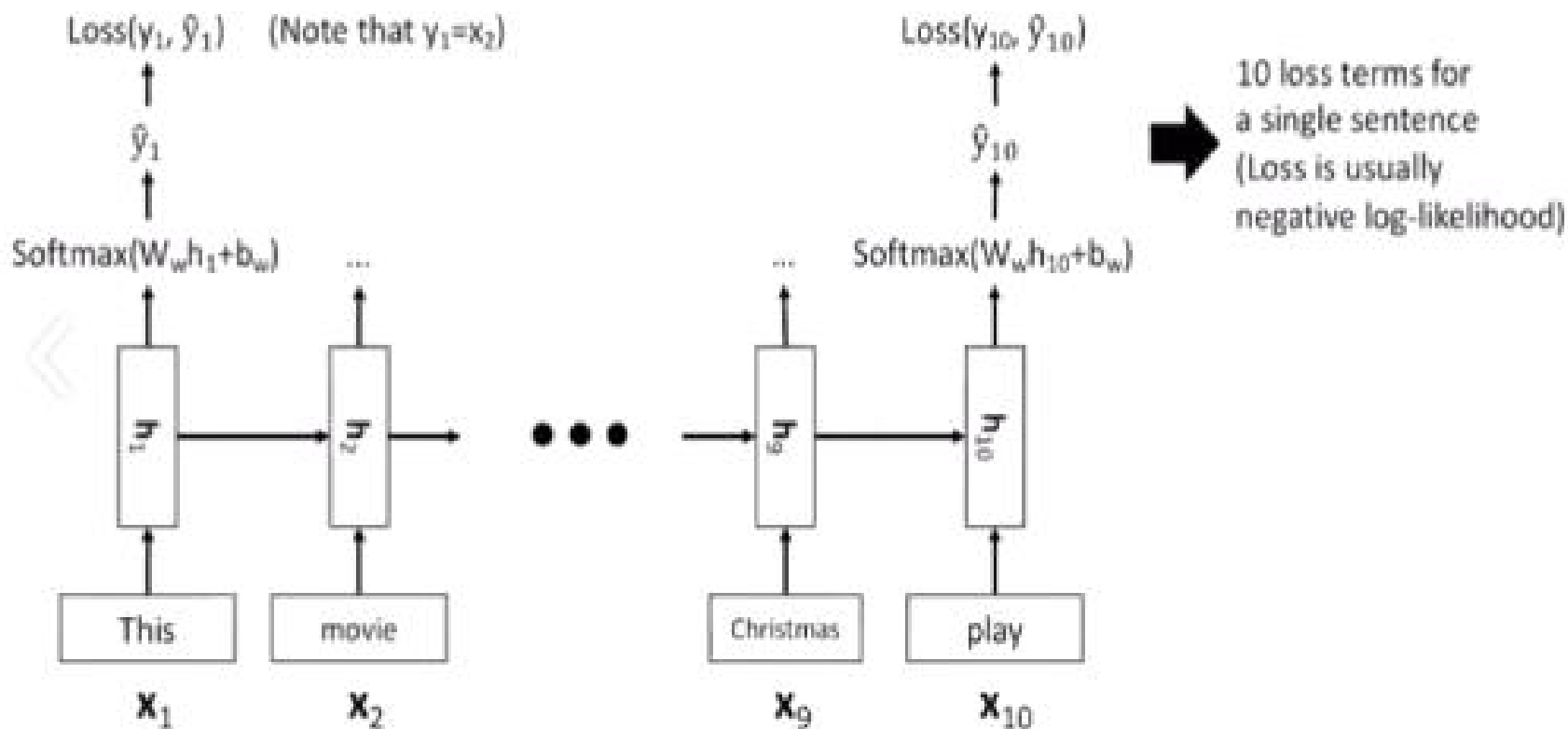
"This movie is as impressive as a preschool Christmas play" → 긍정?(1) 부정?(0)



# 모델 설명 - GRU

## Example

"This movie is as impressive as a preschool Christmas play" → 긍정?(1) 부정?(0)



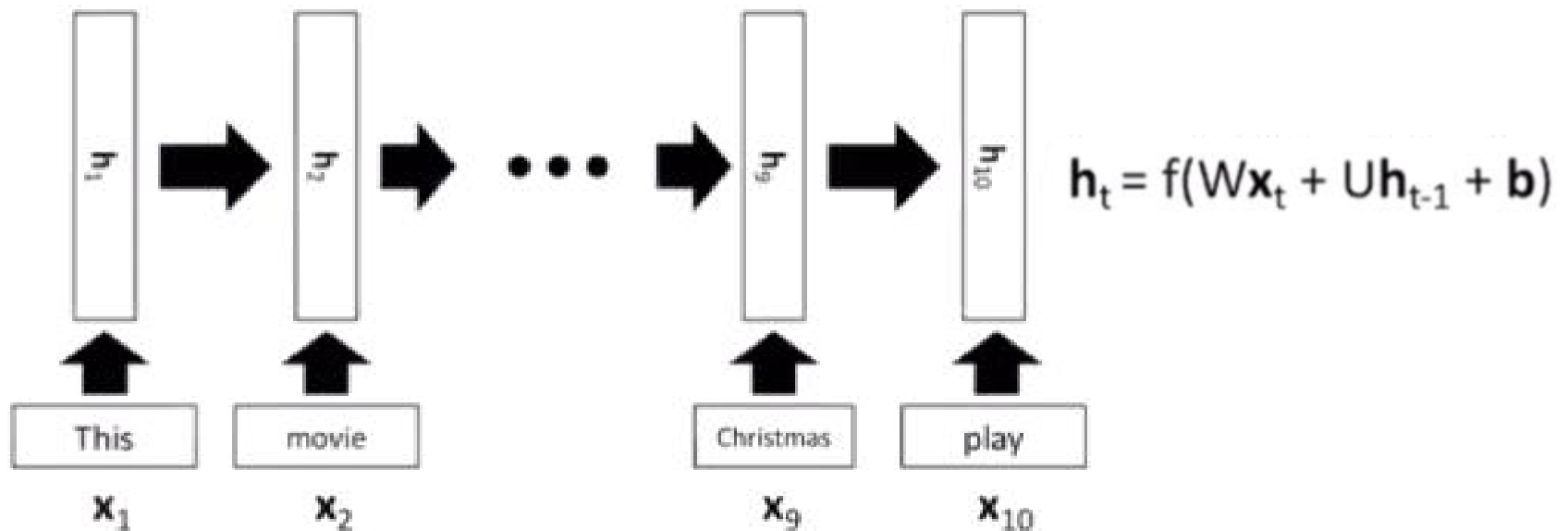
# 모델 설명 - GRU

[RNN의 한계]

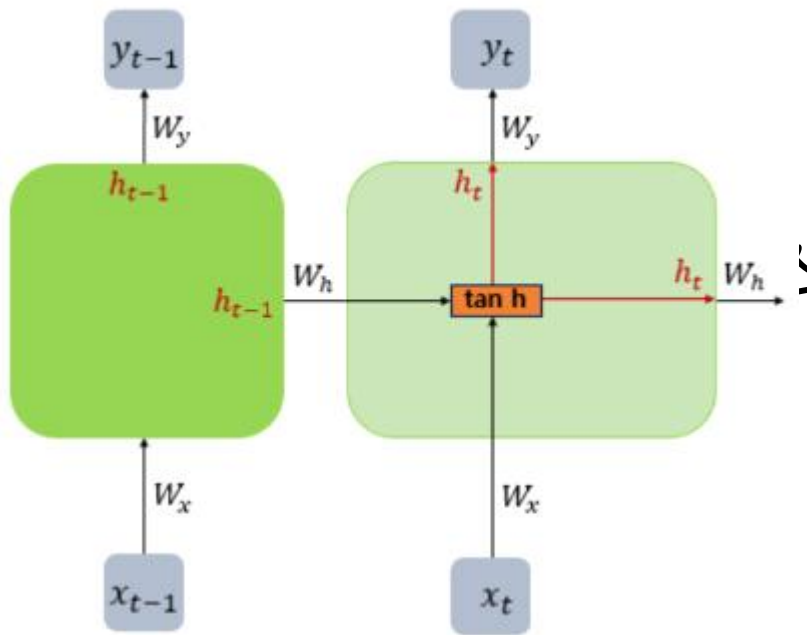
1. 장기 의존성 문제 (Longterm dependency problem)
2. 기울기 소실 (Gradient Vanishing)



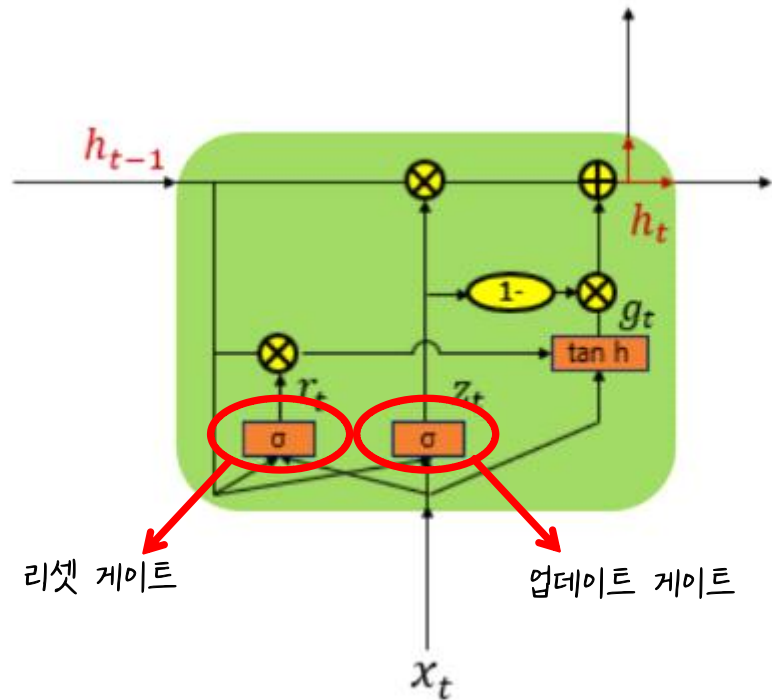
LSTM (Long Short-Term Memory)  
& GRU (Gated Recurrent Unit)



# 모델 설명 - GRU

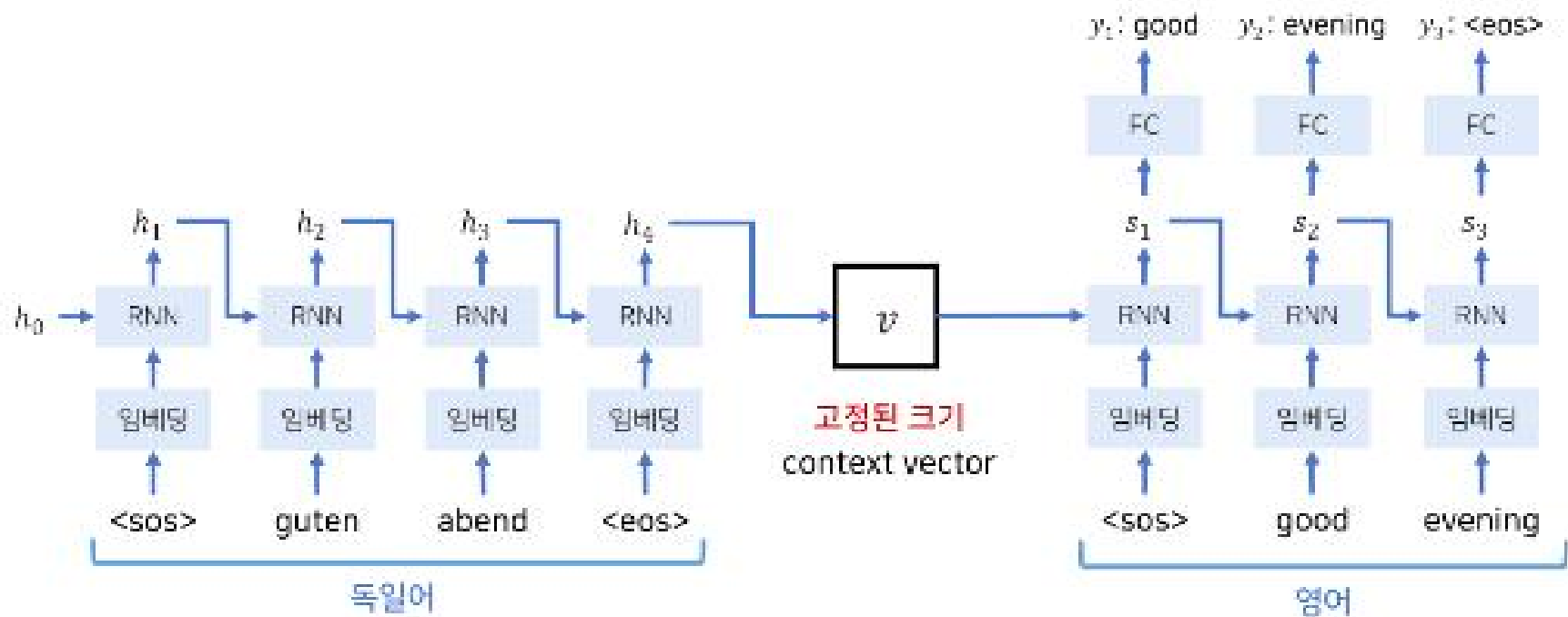


(RNN의 구조)



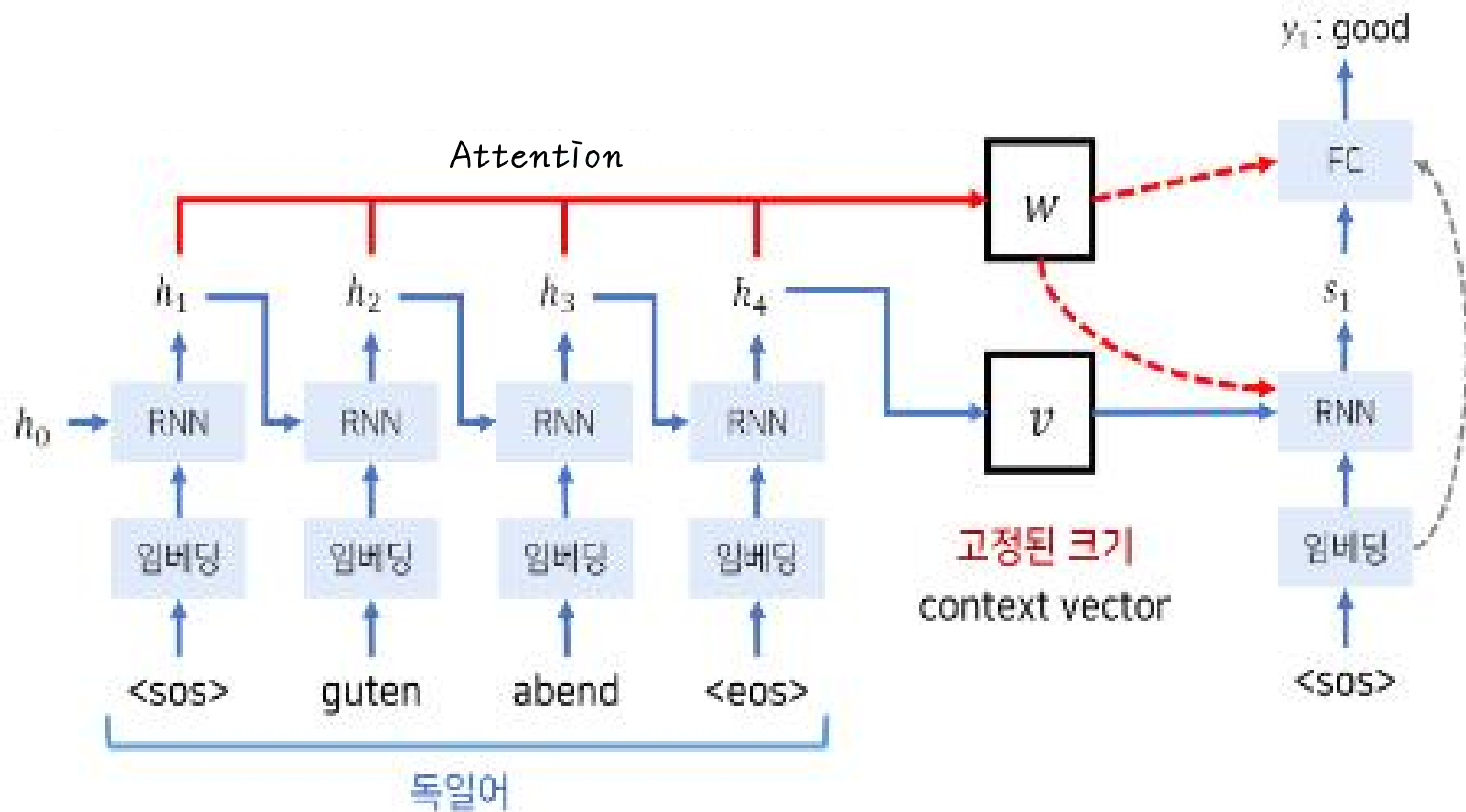
(GRU의 구조)

# 모델 설명 - Transformer



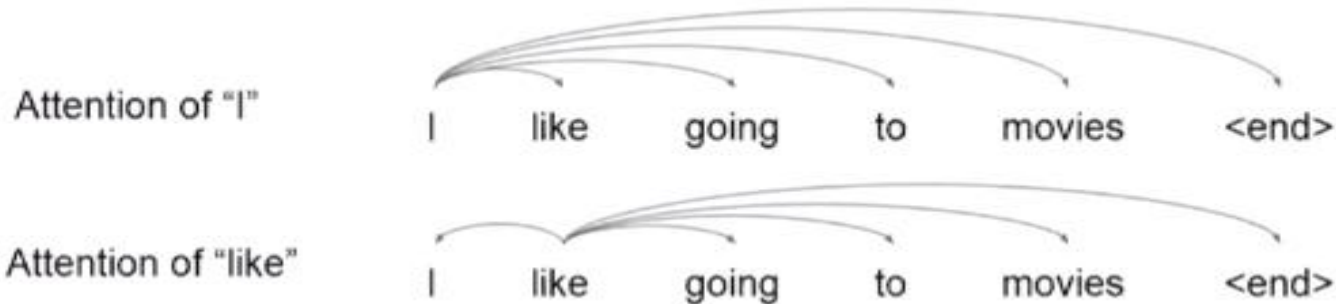
[RNN 기반(LSTM, GRU)의 번역 예시]

# 모델 설명 - Transformer





# 모델 설명 - Transformer



- $Attention(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V$

0.5	0.1	0.0	0.2	0.2	0.0
0.2	0.6	0.0	0.0	0.1	0.0
...	...	...	...	...	...
...	...	...	...	...	...
...	...	...	...	...	...
...	...	...	...	...	...

$$\text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)$$

I
like
going
to
movies
<end>

$V$



0.5*I + 0.1*like + 0.2*to + 0.2* movies
0.2*I + 0.6*like + 0.1*movies
...
...
...
...

$Attention(Q, K, V)$

# 모델 설명 - Transformer

Input

Thinking

Machines

Embedding

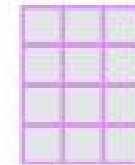
$x_1$  

$x_2$  

Queries

$q_1$  

$q_2$  

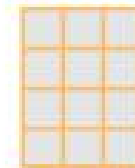


$W^Q$

Keys

$k_1$  

$k_2$  

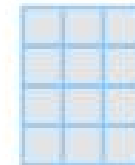


$W^K$

Values

$v_1$  

$v_2$  



$W^V$

# 모델 설명 - Transformer

Input

Thinking

Machines

Embedding

$x_1$



$x_2$



Queries

$q_1$



$q_2$



Keys

$k_1$



$k_2$

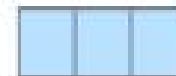


Values

$v_1$



$v_2$



Score

$$q_1 \cdot k_1 = 112$$

$$q_1 \cdot k_2 = 96$$

# 모델 설명 - Transformer

Input

Embedding

Queries

Keys

Values

Score

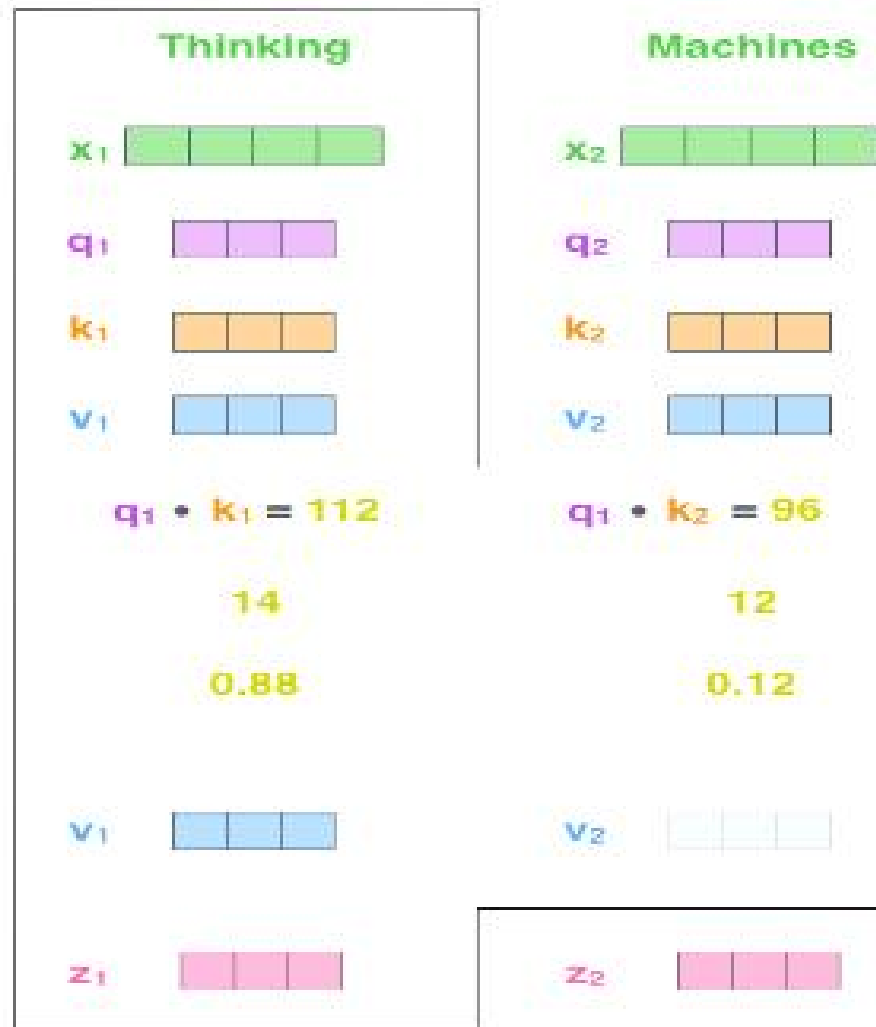
Divide by 8 (  $\sqrt{d_k}$  )

Softmax

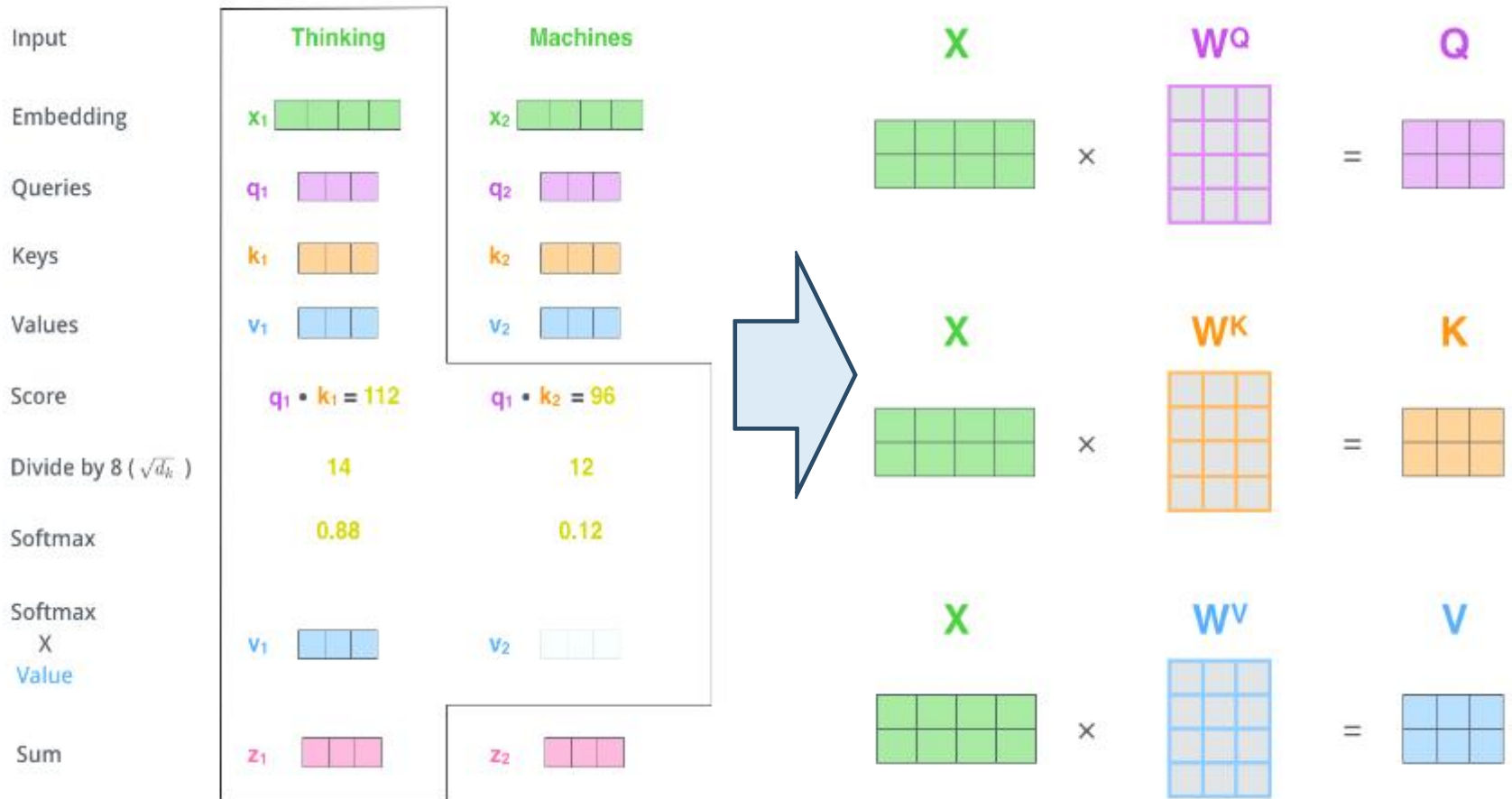
Softmax

X  
Value

Sum



# 모델 설명 - Transformer



# 모델 설명 - Transformer

- $Attention(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V$

0.5	0.1	0.0	0.2	0.2	0.0
0.2	0.6	0.0	0.0	0.1	0.0
...	...	...	...	...	...
...	...	...	...	...	...
...	...	...	...	...	...
...	...	...	...	...	...

$$\text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)$$

I  
like  
going  
to  
movies  
<end>

V



0.5*I + 0.1*like + 0.2*to + 0.2* movies
0.2*I + 0.6*like + 0.1*movies
...
...
...
...

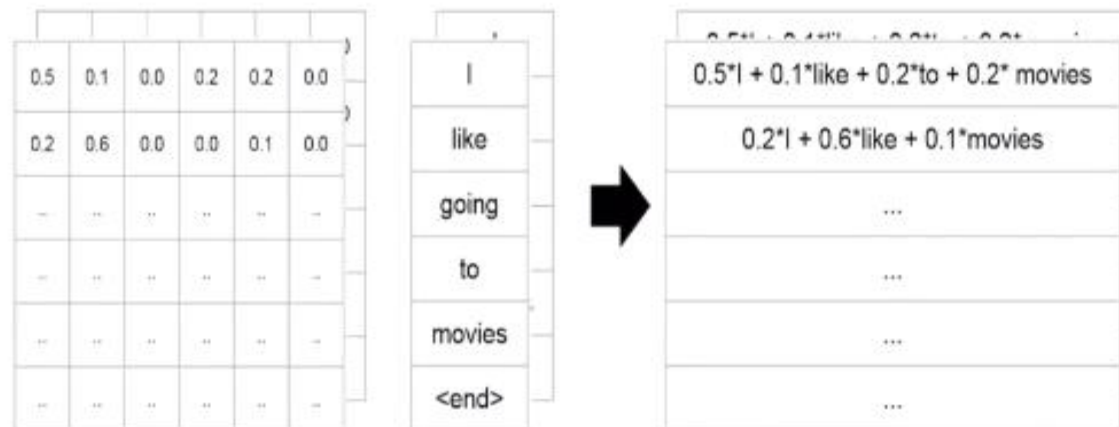
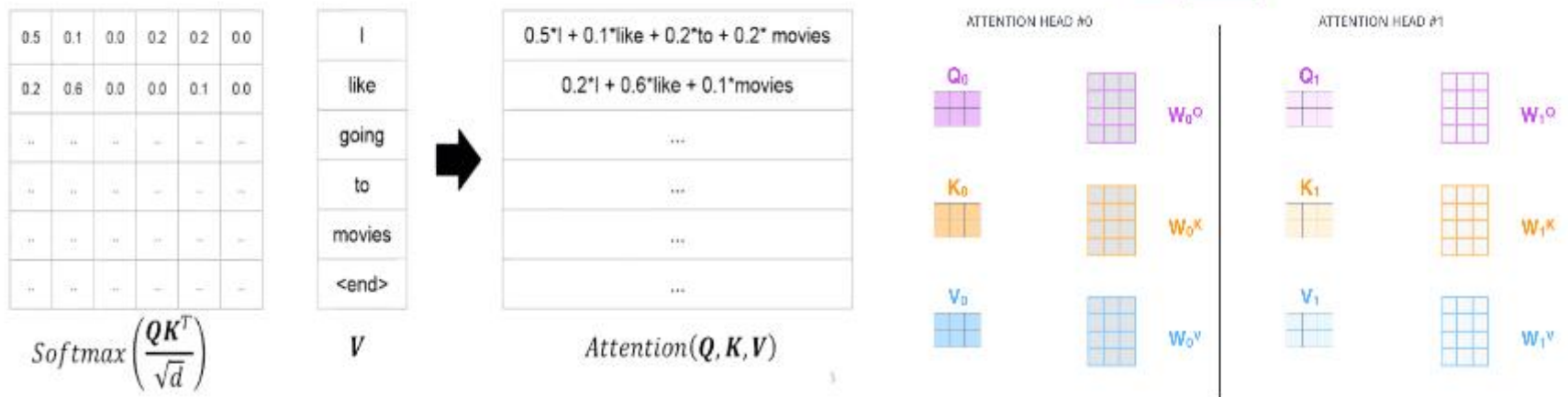
$$Attention(Q, K, V)$$

$$\text{softmax}\left(\frac{\begin{matrix} \text{Q} \\ \text{2x3 grid} \end{matrix} \times \begin{matrix} \text{K}^T \\ \text{3x2 grid} \end{matrix}}{\sqrt{d_k}}\right) \begin{matrix} \text{V} \\ \text{3x2 grid} \end{matrix}$$

$$= \begin{matrix} \text{Z} \\ \text{2x3 grid} \end{matrix}$$

# 모델 설명 - Transformer

- $Attention(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V$



# 모델 설명 - Transformer

X  
Thinking  
Machines



Calculating attention separately in  
eight different attention heads


ATTENTION  
HEAD #0

$Z_0$



ATTENTION  
HEAD #1

$Z_1$



...

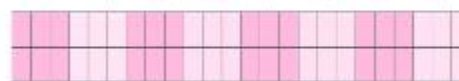
ATTENTION  
HEAD #7

$Z_7$



1) Concatenate all the attention heads

$Z_0$   $Z_1$   $Z_2$   $Z_3$   $Z_4$   $Z_5$   $Z_6$   $Z_7$




2) Multiply with a weight  
matrix  $W^O$  that was trained  
jointly with the model

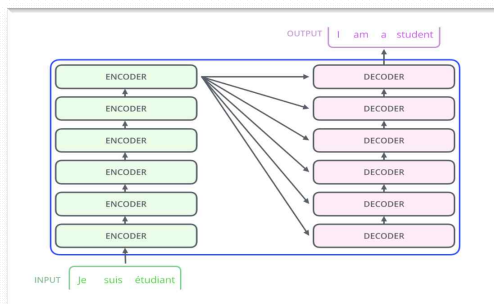
X

3) The result would be the  $Z$  matrix that captures information  
from all the attention heads. We can send this forward to the FFNN

$Z$



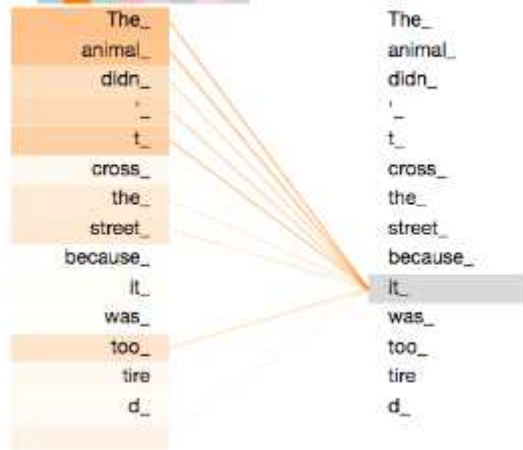
$W^O$



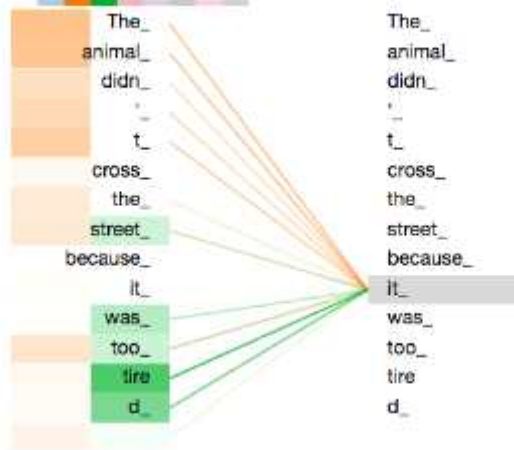
# 모델 설명 - Transformer

Layer: 5 Attention: Input - Input



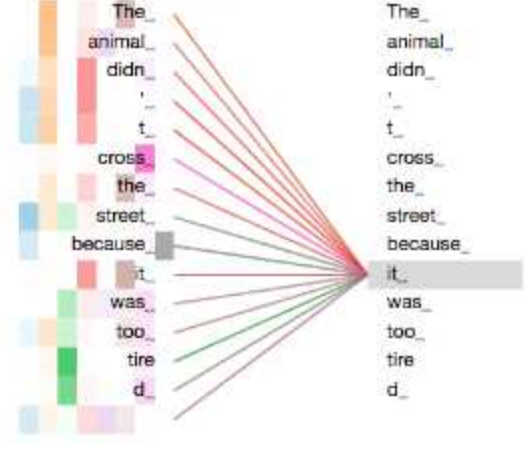
1 attention

Layer: 5 Attention: Input - Input



2 attentions

Layer: 5 Attention: Input - Input



8 attentions

# 모델 설명 - Transformer

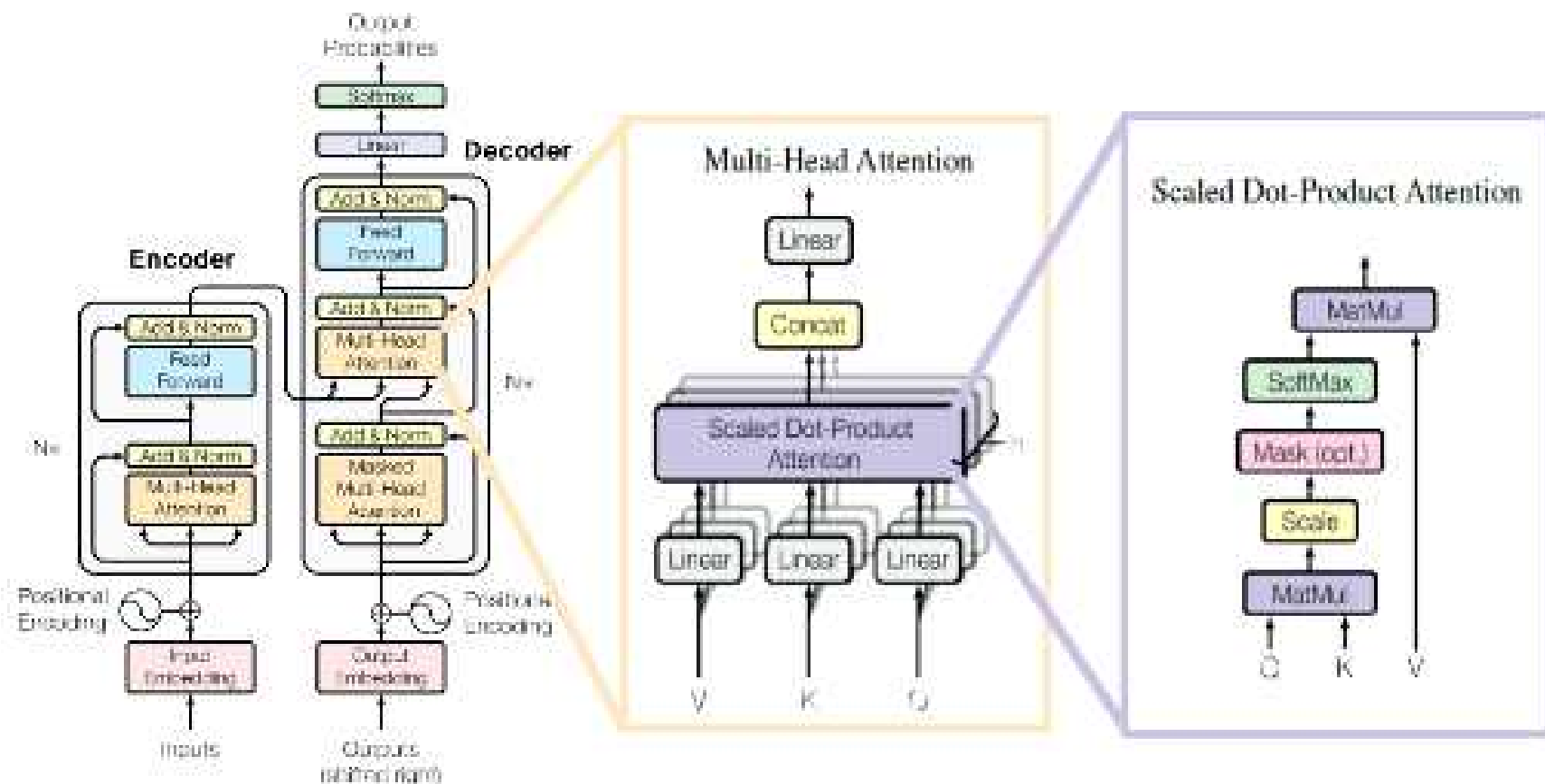
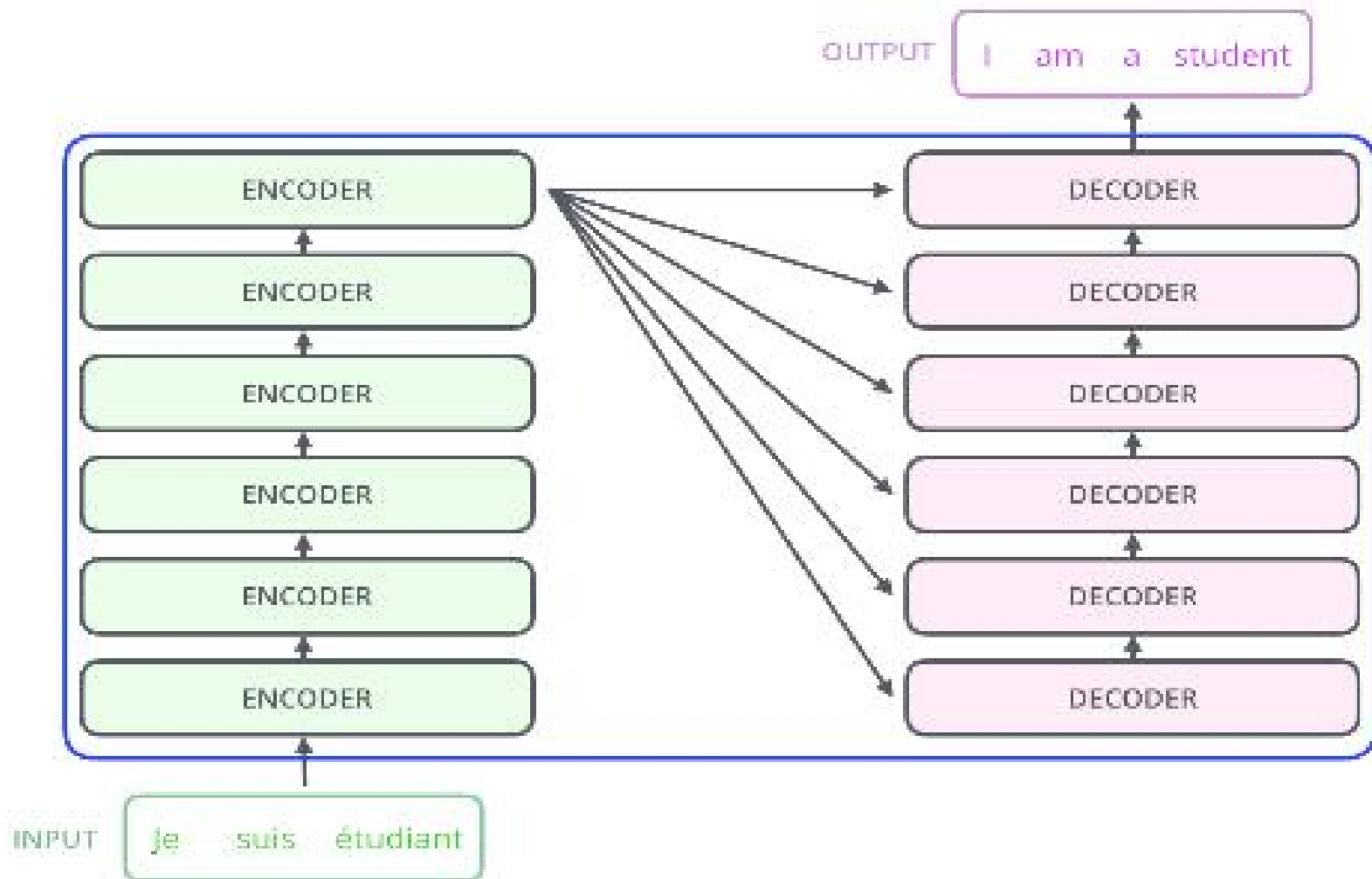
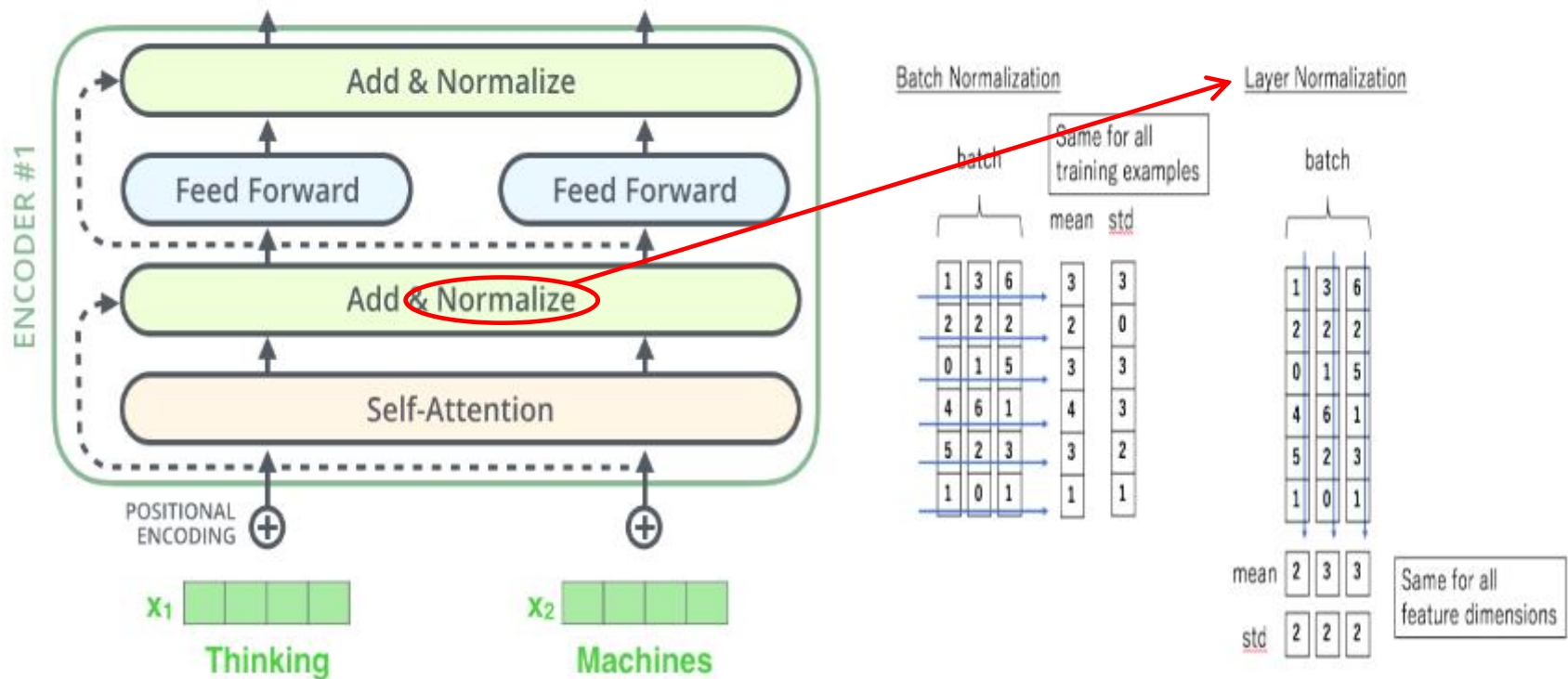


Figure 1: The Transformer - model architecture.

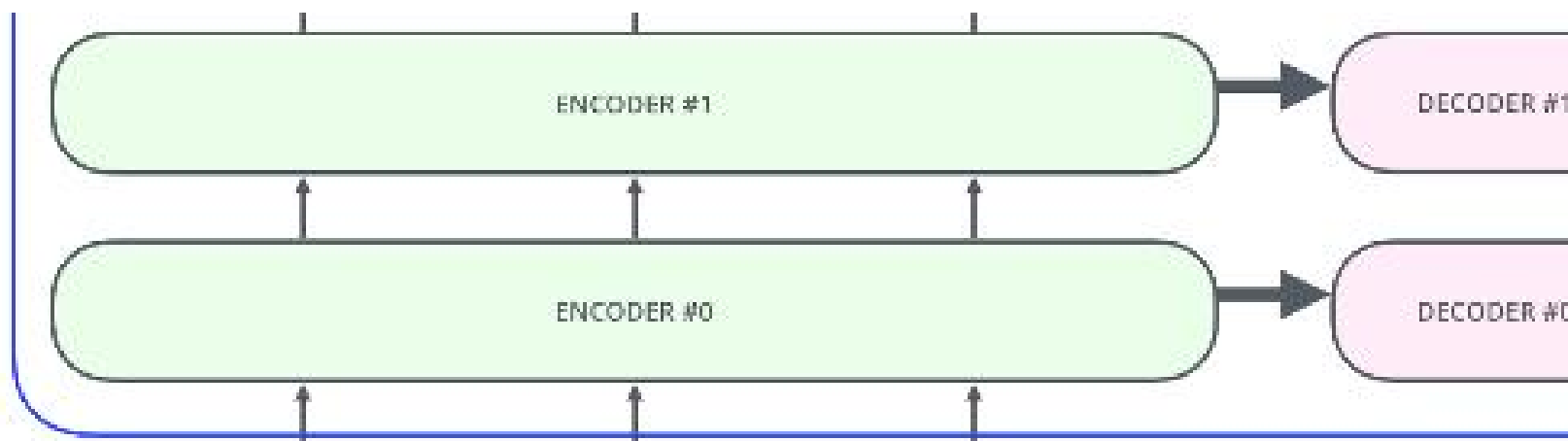
# 모델 설명 - Transformer



# 모델 설명 - Transformer



# 모델 설명 - Transformer



EMBEDDING  
WITH TIME  
SIGNAL

$x_1$

$x_2$

$x_3$

=

=

=

POSITIONAL  
ENCODING

$t_1$

$t_2$

$t_3$

+

+

+

EMBEDDINGS

$x_1$

$x_2$

$x_3$

INPUT

Je

suis

étudiant

# 모델 설명 - Transformer

<How to Positional Encoding>

1. 각 time-step(문장에서 단어의 위치)에 대해 고유한 인코딩을 출력해야 합니다.
2. 일정 time-step 사이의 거리는 길이가 다른 문장끼리 일정해야 한다.
3. test data에서 train data에서보다 더 긴 시퀀스가 들어왔을때도 처리할 수 있어야 한다. (더 긴 문장도 일반화가 가능해야함)  
→ 상한(upper bound)이 필요하다.
4. 정확하게 위치를 결정할 수 있어야 한다.(확률값 x)

## • Training sample

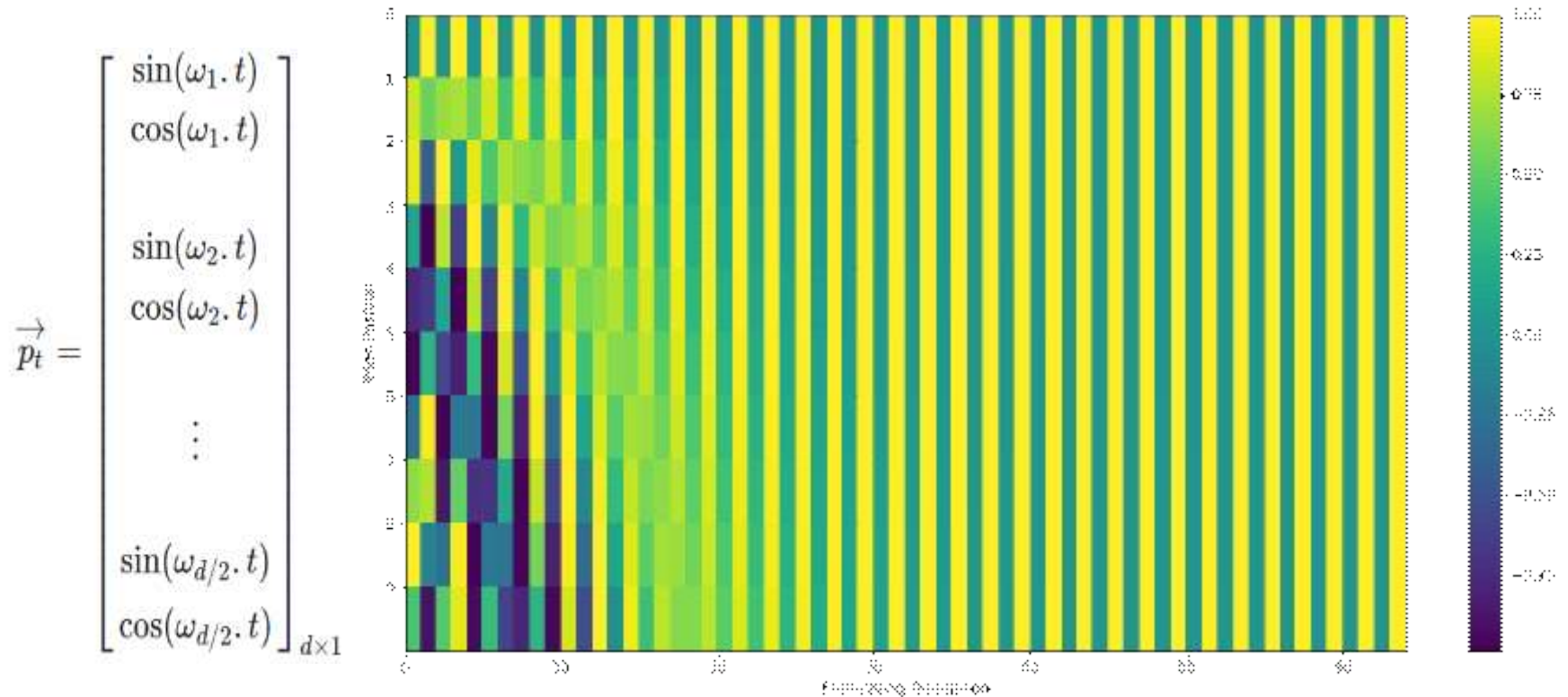
Sentence	Dark	horses	are	faster	than	white	horses
Pos 1	1	2	3	4	5	6	7
Pos 2	0.14	0.28	0.42	0.56	0.70	0.84	1.00

## • Test sample

Sentence	Dark	horses	might	be	faster	than	white	horses
Pos 1	1	2	3	4	5	6	7	8
Pos 2	0.12	0.25	0.37	0.50	0.62	0.75	0.87	1.00

# 모델 설명 - Transformer

$$\vec{p}_t^{(i)} = f(t)^{(i)} := \begin{cases} \sin(\omega_k \cdot t), & \text{if } i = 2k \\ \cos(\omega_k \cdot t), & \text{if } i = 2k + 1 \end{cases} \quad \omega_k = \frac{1}{10000^{2k/d}}$$



# 모델 설명 - Transformer

$$\vec{p_t} = \begin{bmatrix} \sin(\omega_1 \cdot t) \\ \cos(\omega_1 \cdot t) \\ \\ \sin(\omega_2 \cdot t) \\ \cos(\omega_2 \cdot t) \\ \\ \vdots \\ \\ \sin(\omega_{d/2} \cdot t) \\ \cos(\omega_{d/2} \cdot t) \end{bmatrix}_{d \times 1}$$

```
import math
import matplotlib.pyplot as plt

n = 4 # 단어(word)의 개수
dim = 8 # 임베딩(embedding) 차원

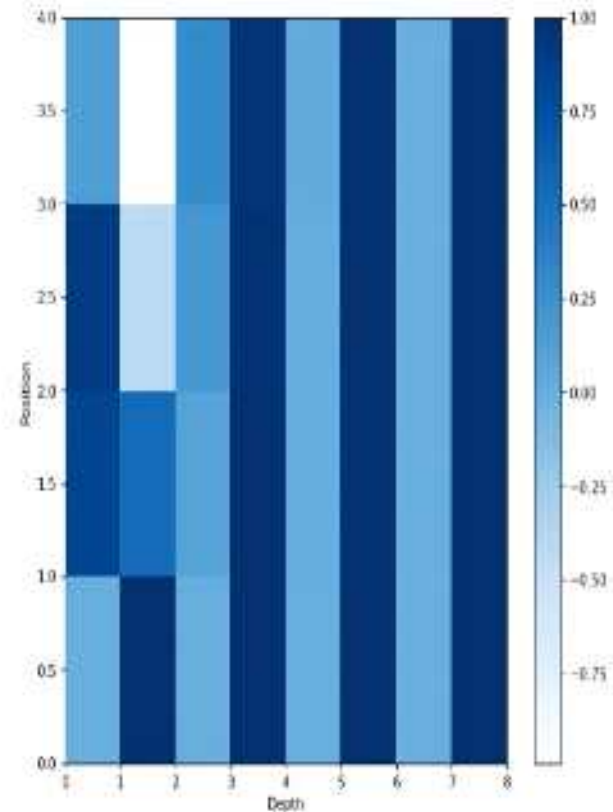
def get_angles(pos, i, dim):
    angles = 1 / math.pow(10000, (2 * (i // 2)) / dim)
    return pos * angles

def get_positional_encoding(pos, i, dim):
    if i % 2 == 0: # 짝수인 경우 사인 함수
        return math.sin(get_angles(pos, i, dim))
    # 홀수인 경우 코사인 함수
    return math.cos(get_angles(pos, i, dim))

result = [[0] * dim for _ in range(n)]

for i in range(n):
    for j in range(dim):
        result[i][j] = get_positional_encoding(i, j, dim)
```

출력 결과: plt.pcolormesh(result, cmap='Blues')





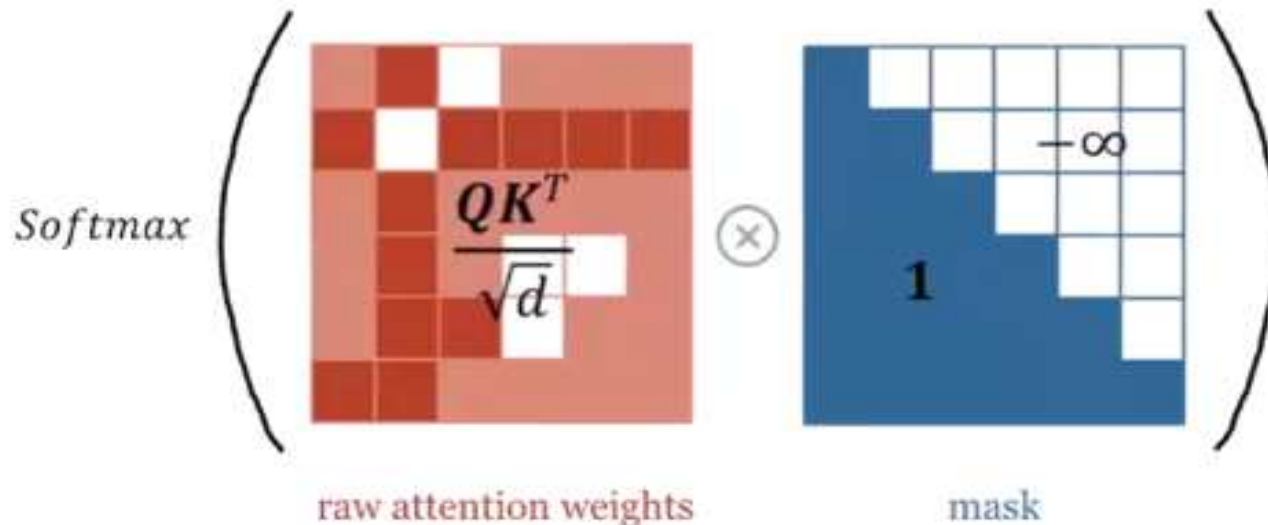
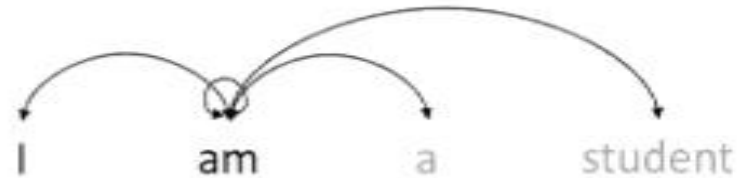
# 모델 설명 - Transformer

## Masked Multi-Head Attention

Encoder attention from "suis"



Decoder attention from "am"



# 모델 설명 - Transformer

## Masked Multi-Head Attention

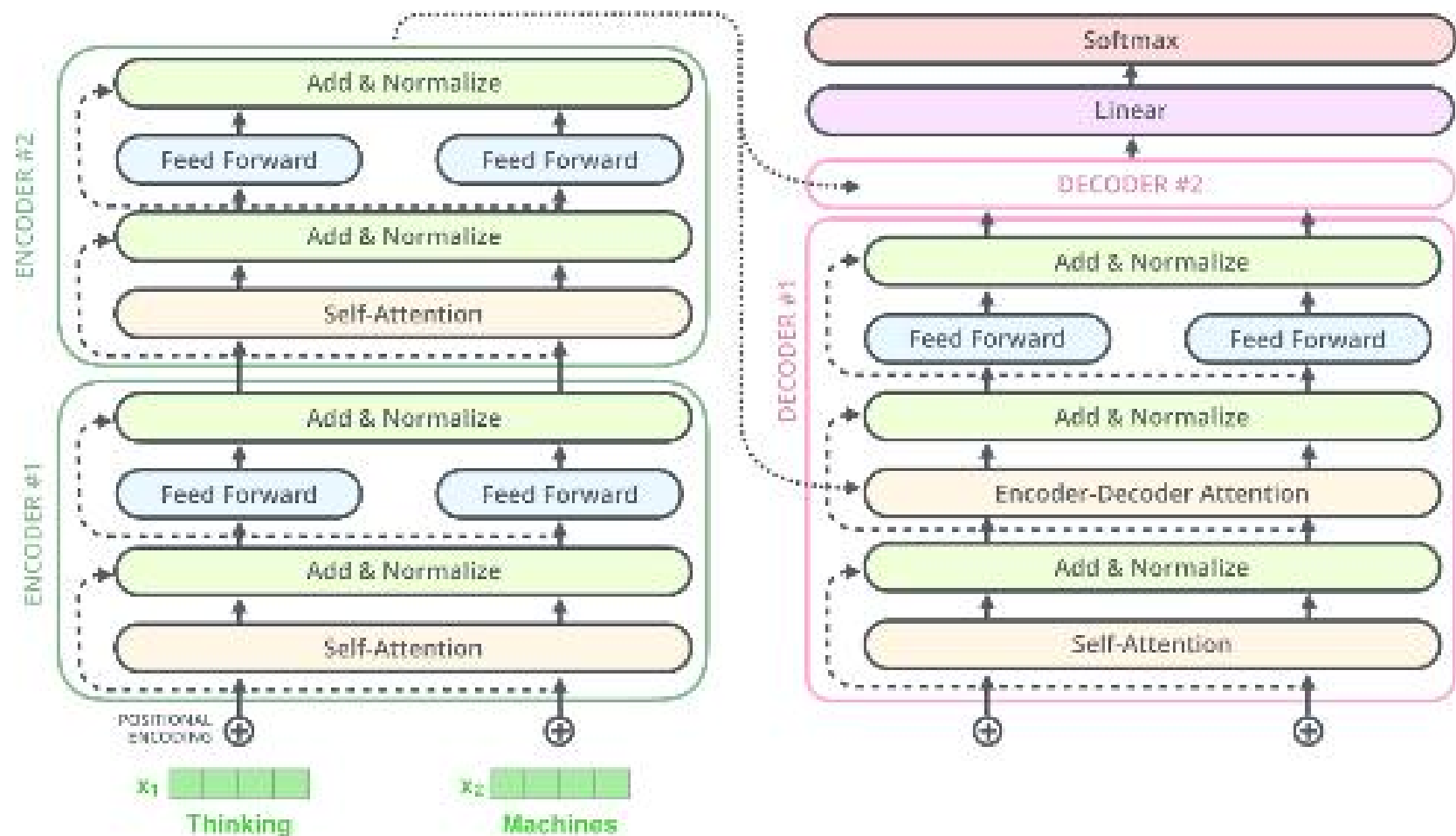
1.0	0	0	0	0	0
0.3	0.7	0	0	0	0
0.2	0.3	0.5	0	0	0
0.1	0.2	0.1	0.6	0	0
...	...	...	...	...	0
...	...	...	...	...	...

I
like
going
to
movies
.



$1.0 * I$
$0.3 * I + 0.7 * \text{like}$
$0.2 * I + 0.3 * \text{like} + 0.5 * \text{going}$
$0.1 * I + 0.2 * \text{like} + 0.1 * \text{going} + 0.6 * \text{to}$
...
...

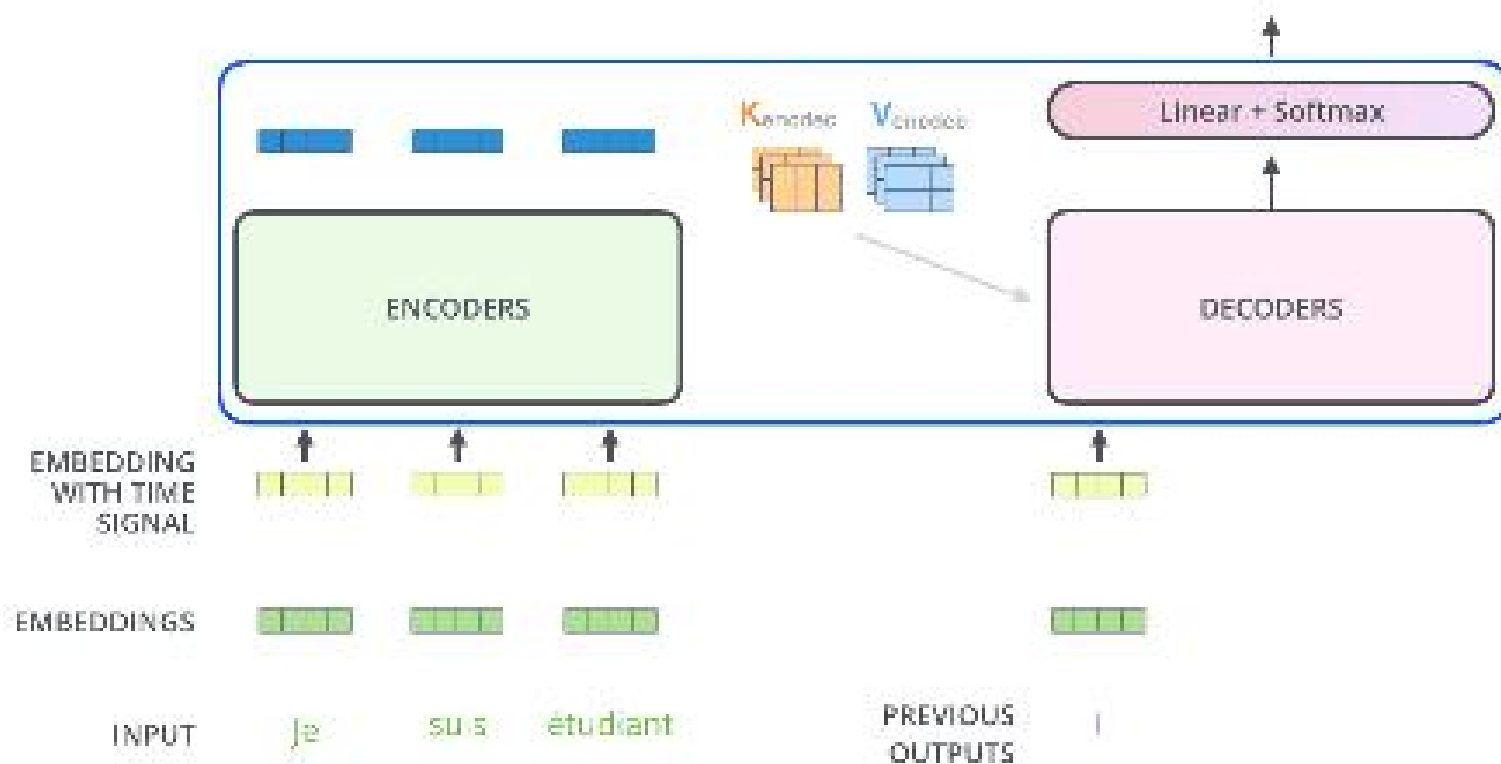
# 모델 설명 - Transformer



# 모델 설명 - Transformer

Decoding time step: 1 2 3 4 5 6

OUTPUT



# 목차

## Table of contents



NLP(Natural Language Processing)



Language Model (언어모델)



NLP의 활용 (감성분석 예시)



코드 설명



차이점 및 개선사항



NLP와 통계학

# 코드 설명 - 크롤러

<주요 사용 라이브러리>

BeautifulSoup

selenium

pandas

time

openpyxl

```
for i in range(1,945):
    time.sleep(0.5)
    try:
        # javascript page_link('2')
        md.execute_script(f'page_link({i})')
        time.sleep(0.5)
        html = md.page_source
        soup = BeautifulSoup(html, 'html.parser')
        notices = soup.select('#menu2021_cbl16 > div._fnctWrap._articleT')
        # print(notices)
        notice = notices.select('tr')
        for a in notice[2:]:
            index = a.select_one("._articleTdNum").text.strip()
            # print(index)
            title = a.select_one("._articleLinkView").text.strip()
            # print(title)
            writer = a.select_one("._articleTdWriter").text.strip()
            # print(writer)
            date = a.select_one("._articleTdRdate").text.strip()
            # print(date)
            upNum = a.select_one("._articleTdUp").text.strip()
            # print(upNum)
            downNum = a.select_one("._articleTdDown").text.strip()
            # print(downNum)
            view = a.select_one("._articleTdAccess").text.strip()
            # print(index)
            # print([index, title, writer, date, upNum, downNum, view])
            sheet.append([index, title, writer, date, upNum, downNum, view])
    except:
        count += 1
        continue

if i % 30 == 0:
    print(i, "페이지까지 크롤링 완료, 현재 오류 ", count, "번")
    wb.save("inha_plaza.xlsx")
    print("크롤링 완료, 오류 총 ", count, "번")
```

번호	종류	제목	작성자	작성일	업	다운	조회수
13901	발행	[인공지능융합연구센터] 교정서 0211 세제 신청 안내(장사 접수)	한미현	2022.06.29	0	0	121
13902	모집	[00독일공보출판센터] 2022학년도 2학기 [00]인공지능실습 참여확산 모집 안내	리다경	2022.05.10	4	1	400
13903	모집	인위대학과 교수학습개발센터 전문연구원 채용	단건터	2022.07.26	0	0	17
13904	발행	유유 세공 이벤트! 인위인 비전공자를 위한 유유 컷집착도 인도가 간외	김지훈	2022.07.26	1	0	38
13905	보급	[가정경제 청년기초지원단] 교육사업(청년기초) 참여 희망 수요 설문조사(2022년 3차 모집대상) <2022.7.06.06>	이희진	2022.07.25	0	0	34
13906	모집	[인위대명상] 건강한 생활태양자를 모집합니다.	이태진	2022.07.25	0	1	67
13907	모집	[조는 수확지도] 조는 수확지도 신청날 구합니다.	고복경	2022.07.24	0	0	108
13908	발행	[사범대학 캠퍼스별별교외] 2022-1학기 사범대학 정기캠퍼스 일과 공고	김종영	2022.07.24	1	0	62
13909	발행	[아이다이벤트]교정생자한은 최정자 아이다이벤트 참여인들만 가능함이다	이종화	2022.07.23	13	1	161
13910	모집	[인위대학생회] 스마트디지털센터 [어린친구] 부진금 내화양반 모집	주호영	2022.07.22	0	0	15
13911	발행	[열문지] [인위대신문] 100이 기념 '인위대확신문'은 없다! 열문지사 <수신을 통해 신문 검증> <09.01>	이재원	2022.07.22	4	0	50
13912	모집	[인위대신문] 미디어콘텐츠 기자교 모집합니다. [모집기간] 2022.07.20일(제1차)	이재원	2022.07.22	2	0	34
13913	호외	[호외(1) 큐브네] 비외 수확지도 운영 안내	이재원	2022.07.22	4	0	160
13914	모집	[미래사육자지원] 2022 학제별인정기초 skh up 레노 키: 에노비노를 인문학 전공수행 논문 교육 및 미디어	이연지	2022.07.21	1	0	65
13915	모집	[미래지능사육자지원] 제2차 학제별인정기초 테라노스 콘테스트	이연지	2022.07.21	1	0	32
13916	발행	[최정TT] 2차 가제 세무 가이드라인 안내	김준연	2022.07.21	6	1	294
13917	발행	[인위대신문] 7월 21일 주권 안내	한미현	2022.07.21	0	2	94

# 코드 설명 - 감성 분석 (라벨링)

<주요 사용 라이브러리>

numpy  
pandas  
torch  
torchtext  
gensim  
konlpy

[NLP Process]

데이터 라벨링 -> 데이터 전처리 (불용어처리 및 형태소 분해) -> train, val, test data 나누기  
-> word2vec 변환(word embedding) -> 모델 생성 및 하이퍼파라미터 설정 -> 학습 -> 평가

in [12]: df

Out [12]:

	Unnamed: 0		title	upNum	downNum
0	14155		[계절, 학기, 등록금, 생활, 왜, 많이]	29	7
1	14149		[인하대병원, 사무, 행정, 직, 인턴, 사원, 차등, 진료료]	5	0
2	14149		[물, 인하, 누리, 학년, 도제, 학기, 생활관, 모집, 안내]	1	0
3	14150		[방학, 동안, 학생, 후기, 라문]	1	0
4	14154		[연구실, 신입생, 및, 인턴, 모집]	4	0
...	...		...	...	...
14147	8		[상담, 센터, 학년, 도제, 생활, 집단, 상담, 신청, 안내]	3	0
14150	5		[년, 학기, 연구실, 안전교육, 이수자, 커피, 푸른, 행정, 학생, 안내]	1	0
14152	3		[인하, 대학교, 인공, 지능, 및, 최적화, 연구실, 초청회, 박사, 과정, 영...	2	0
14153	2		[극비, 지원, 카타르피공, 면접, 극비, 모집]	1	0
14154	1		[인하, 청년, 센터, 마루, 취업, 특강, 모습, 기초, 공학, 법, 모집, 안내]	1	0

9753 rows x 4 columns



```
In [13]: df['label'] = np.where(df['upNum'] > df['downNum'], 1, 0)
```

```
In [14]: df = df.drop('upNum', axis=1)
df = df.drop('downNum', axis=1)
```

```
In [15]: df
```

```
Out [15]:
```

	Unnamed: 0		title	label
0	14155		[계절, 학기, 등록금, 생활, 왜, 많이]	1
1	14149		[인하대병원, 사무, 행정, 직, 인턴, 사원, 차등, 진료료]	1
2	14149		[물, 인하, 누리, 학년, 도제, 학기, 생활관, 모집, 안내]	1
3	14150		[방학, 동안, 학생, 후기, 라문]	1
4	14154		[연구실, 신입생, 및, 인턴, 모집]	1
...	...		...	...
14147	8		[상담, 센터, 학년, 도제, 생활, 집단, 상담, 신청, 안내]	1
14150	5		[년, 학기, 연구실, 안전교육, 이수자, 커피, 푸른, 행정, 학생, 안내]	1
14152	3		[인하, 대학교, 인공, 지능, 및, 최적화, 연구실, 초청회, 박사, 과정, 영...	1
14153	2		[극비, 지원, 카타르피공, 면접, 극비, 모집]	1
14154	1		[인하, 청년, 센터, 마루, 취업, 특강, 모습, 기초, 공학, 법, 모집, 안내]	1

9753 rows x 3 columns

```
In [15]: df['label'].value_counts()
```

```
Out [15]:
```

```
1    9098
0    1655
Name: label, dtype: int64
```

# 코드 설명 - 감성 분석 (전처리)

<주요 사용 라이브러리>

numpy  
pandas  
torch  
torchtext  
gensim  
konlpy

## [NLP Process]

데이터 라벨링 -> 데이터 전처리 (불용어처리 및 형태소 분해) -> train, val, test data 나누기  
-> word2vec 변환(word embedding) -> 모델 생성 및 하이퍼파라미터 설정 -> 학습 -> 평가

```
In [21]: from konlpy.tag import Okt
          okt = Okt()

          def tokenizer_Okt(text):
              return [tok for tok in okt.nouns(text)]
```

```
In [22]: import re

          def preprocessing_text(text):
              text = re.sub('[\r\n]', '', str(text))
              text = re.sub('[\n]', '', str(text))
              text = re.sub('[\t]', '', str(text))
              text = re.sub('[\f]', '', str(text))
              text = re.sub('[\r]', '', str(text))
              text = re.sub('[\f]', '', str(text))

              # 숫자를 왼쪽으로 '0'로 설정
              text = re.sub(r'[0-9 0-9]', ' ', str(text)) # 숫자

          return text
```

```
In [23]: # 전처리 및 okt 단어 분해를 수행하는 함수를 정의한다
          def tokenizer_with_preprocessing(text):
              text = preprocessing_text(text) # 전처리 정규화
              ret = tokenizer_Okt(text) # okt 단어 분해

          return ret

          # 동작 확인
          test_text = "★일정 &진명 동문회 신입생 모집★"
          print(tokenizer_with_preprocessing(test_text))

          ['일정', '진명', '동문회', '신입생', '모집']
```

```
In [24]: from torchtext.legacy.data import Field, TabularDataset

          # csv 파일을 읽을 때, 필드들의 내용에 대해 수행할 처리를 정의합니다
          # 공백과 라벨을 모두 분리합니다

          max_length = 25
          TEXT = Field(sequential=True, tokenize=tokenizer_with_preprocessing,
                      use_vocab=True, lower=True, include_lengths=True, batch_first=True, fix_length=max_length)
          LABEL = Field(sequential=False, use_vocab=False)

          # 함수의 의미는 다음과 같습니다
          # sequential: 단어의 길이가 가변인가? 문장은 길이가 다양하므로 True, 라벨은 False
          # tokenize: 문장을 읽을 때, 전처리 및 단어 분해 함수를 정의
          # use_vocab: 단어를 vocabulary(단어리: 이후에 설명)에 추가할지 여부
          # lower: 알파벳이 존재할 때 소문자로 변환할지 여부
          # include_length: 문장의 단어 수 데이터를 포함할지 여부
          # batch_first: mini batch 처리를 선두에 제공할지 여부
          # fix_length: 전정 문장을 지정된 길이로 맞추기 위하여 padding
```



<주요 사용 라이브러리>

numpy  
pandas  
torch  
torchtext  
gensim  
konlpy

# 코드 설명 - 감성 분석 (data split)

## [NLP Process]

데이터 라벨링 -> 데이터 전처리 (불용어처리 및 형태소 분해) -> train, val, test data 나누기  
-> word2vec 변환 (word embedding) -> 모델 생성 및 하이퍼파라미터 설정 -> 학습 -> 평가

```
In [27]: train_df = train_df.drop('Unnamed: 0', axis=1)
         val_df = val_df.drop('Unnamed: 0', axis=1)
         test_df = test_df.drop('Unnamed: 0', axis=1)
```

```
In [28]: train_df.to_csv('data/train.csv', index=False)
         val_df.to_csv('data/val.csv', index=False)
         test_df.to_csv('data/test.csv', index=False)
```

```
In [29]: from torchtext.legacy.data import TabularDataset

         train, validation, test = TabularDataset.splits(
             path = 'data/',
             train = 'train.csv',
             validation = 'val.csv',
             test = 'test.csv',
             format = 'csv',
             fields = [('text', TEXT), ('label', LABEL)],
             skip_header = True
         )

         print("Train:", train[0].text, train[0].label)
         print("Validation:", validation[0].text, validation[0].label)
         print("Test:", test[0].text, test[0].label)
```

```
Train: ['기숙사', '남자', '문구'] 1
Validation: ['총학생회', '실종', '요'] 1
Test: ['출문위', '제', '중앙', '운영', '위원회', '차', '회의', '결과', '보고'] 0
```

# 코드 설명 - 감성 분석 (embedding)

<주요 사용 라이브러리>

numpy  
pandas  
torch  
torchtext  
gensim  
konlpy

## [NLP Process]

데이터 라벨링 → 데이터 전처리 (불용어처리 및 형태소 분해) → train, val, test data 나누기  
→ word2vec 변환 (word embedding) → 모델 생성 및 하이퍼파라미터 설정 → 학습 → 평가

```
from gensim.models import Word2Vec

embedding_model = Word2Vec(train_df['title'],
                           sa = 1, # skip gram
                           min_count = 4,
                           min_count = 1,
                           workers = 4
                           )

print(embedding_model)

model_result = embedding_model.mv.most_similar('[[책]')
print(model_result)

# Word2Vec(vocab=5766, vector_size=100, alpha=0.025)
[('야구', 0.9423664531651306), ('배구', 0.9307630726051331), ('리그', 0.9236006306369561), ('경기', 0.9277569191360474), ('부', 0.9132975838853629), ('전국', 0.911598645314624), ('전구', 0.89033306184363), ('씨름', 0.877577662467865), ('대륙', 0.875062372076416), ('선수권 대회', 0.86639034945755)]
```

```
from gensim.models import KeyedVectors

embedding_model.mv.save_word2vec_format('data/tokens_v2v') # 모델 저장
loaded_model = KeyedVectors.load_word2vec_format('data/tokens_v2v') # 모델 로드

model_result = loaded_model.most_similar('[[책]')
print(model_result)

[('야구', 0.9423664531651306), ('배구', 0.9307630726051331), ('리그', 0.9236006306369561), ('경기', 0.9277569191360474), ('부', 0.9132975838853629), ('전국', 0.911598645314624), ('전구', 0.89033306184363), ('씨름', 0.877577662467865), ('대륙', 0.875062372076416), ('선수권 대회', 0.86639034945755)]
```

# 코드 설명 - 감성 분석 (Model)

## (GRU)

<주요 사용 라이브러리>

numpy  
pandas  
torch  
torchtext  
gensim  
konlpy

### [NLP Process]

데이터 라벨링 → 데이터 전처리 (불용어처리 및 형태소 분해) → train, val, test data 나누기  
→ word2vec 변환 (word embedding) → 모델 생성 및 하이퍼파라미터 설정 → 학습 → 평가

```
In [41]: import os
import torch
import torch.nn as nn
import torch.nn.functional as F
from torchtext import data, datasets
import random
```

```
In [42]: SEED = 5
random.seed(SEED)
torch.manual_seed(SEED)
```

```
Out [42]: <torch._C.Generator at 0x7fc830041b50>
```

```
In [43]: # 하이퍼파라미터
BATCH_SIZE = 8
lr = 0.001
EPOCHS = 10
```

```
class GRU(nn.Module):
    def __init__(self, n_layers, hidden_dim, n_vocab, embed_dim, n_classes, dropout_p=0.2):
        super(GRU, self).__init__()
        self.n_layers = n_layers
        self.hidden_dim = hidden_dim

        self.embed = nn.Embedding(n_vocab+2, embed_dim)
        self.dropout = nn.Dropout(dropout_p)
        # 원래는 batch_first를 True로 했으므로, 여기에서도 batch_first = True로 통일.
        self.gru = nn.GRU(embed_dim, self.hidden_dim,
                           num_layers=self.n_layers,
                           batch_first=True)
        # 최종 은닉층에서, 값을 내기 위한 과정이다.
        self.out = nn.Linear(self.hidden_dim, n_classes)

    def forward(self, x):
        # print(x)
        x = self.embed(x[0])
        # 중요! 최초의 hidden_state를 정의하는 과정을 놓아줘야 한다.
        # 첫번째 hidden_state를 0벡터로 초기화
        h_0 = self._init_state(batch_size=x.size(0))

        # GRU의 리턴값은 (배치 크기, 시퀀스 길이, 은닉 상태의 크기)
        x, _ = self.gru(x, h_0)

        # (배치 크기, 은닉 상태의 크기)의 텐서로 크기가 변경됨. 즉, 마지막 time-step의 은닉 상태를 가져온다.
        h_t = x[:, -1, :]
        self.dropout(h_t)

        # (배치 크기, 은닉 상태의 크기) → (배치 크기, 출력층의 크기)
        logit = self.out(h_t)
        return logit

    def _init_state(self, batch_size=1):
        weight = next(self.parameters()).data
        return weight.new(self.n_layers, batch_size, self.hidden_dim).zero_()
```

데이터 라벨링 -> 더  
-> word2vec 변환

In [531]:

[illegible]

# 코드 설명 - 감성 분석 (train)

<주요 사용 라이브러리>

numpy  
pandas  
torch  
torchtext  
gensim  
konlpy

## [NLP Process]

데이터 라벨링 -> 데이터 전처리 (불용어처리 및 형태소 분해) -> train, val, test data 나누기  
-> word2vec 변환(word embedding) -> 모델 생성 및 하이퍼파라미터 설정 -> 학습 -> 평가

```
In [53]: def train(model, optimizer, train_iter):
    model.train()
    loss_func = nn.CrossEntropyLoss()
    for b, batch in enumerate(train_iter):
        # 배치에서 text와 label을 가져온다.
        x, y = batch.text, batch.label
        # print(x)
        # y.data.sub_(1) # 레이블 값을 0과 1로 변환
        optimizer.zero_grad()

        # 학습 과정을 통해 0, 1 결정, binary0이므로 cross_entropy
        logit = model(x)
        logit = torch.tensor(logit, dtype=torch.float64)
        y = torch.tensor(y, dtype=torch.float64)
        # print(y)
        # print(logit)
        # print(y)
        loss = loss_func(logit, y)
        # print(2)
        # print(loss)
        # back-propagation
        loss.requires_grad_(True)
        loss.backward()
        optimizer.step()
```

```
best_val_loss = None
for e in range(1, EP0CHS+1):
    train(model, optimizer, train_iter)
    val_loss, val_accuracy = evaluate(model, validation_iter)

    print("[Epoch: %d] val loss : %5.2f | val accuracy : %5.2f" % (e, val_loss, val_accuracy))

    # 검증 오차가 가장 작은 최적의 모델을 저장
    if not best_val_loss or val_loss < best_val_loss:
        if not os.path.isdir("snapshot"):
            os.makedirs("snapshot")
        torch.save(model.state_dict(), './snapshot/txtclassification.pt')
        best_val_loss = val_loss
```

```
/opt/conda/lib/python3.7/site-packages/ipykernel_launcher.py:13: UserWarning: To copy construct f
ourceTensor.clone().detach() or sourceTensor.clone().detach().requires_grad_(True), rather than to
del sys.path[0]
/opt/conda/lib/python3.7/site-packages/ipykernel_launcher.py:11: UserWarning: To copy construct f
ourceTensor.clone().detach() or sourceTensor.clone().detach().requires_grad_(True), rather than to
# This is added back by InteractiveShellApp.init_path()
```

```
[Epoch: 1] val loss : 0.30 | val accuracy : 83.03
[Epoch: 2] val loss : 0.30 | val accuracy : 83.03
```

# 코드 설명 - 감성 분석 (eval)

<주요 사용 라이브러리>

numpy  
pandas  
torch  
torchtext  
gensim  
konlpy

## [NLP Process]

데이터 라벨링 -> 데이터 전처리 (불용어처리 및 형태소 분해) -> train, val, test data 나누기  
-> word2vec 변환(word embedding) -> 모델 생성 및 하이퍼파라미터 설정 -> 학습 -> 평가

```
model.load_state_dict(torch.load('./snapshot/txtclassification.pt'))  
test_loss, test_acc = evaluate(model, test_iter)  
print('테스트 오차: %5.2f | 테스트 정확도: %5.2f' % (test_loss, test_acc))
```

# 목차

## Table of contents

□ NLP(Natural Language Processing)

□ Language Model (언어모델)

□ NLP의 활용 (감성분석 예시)

□ 코드 설명

■ 아쉬운점 및 개선사항

□ NLP와 통계학



# 아쉬운 점 및 개선사항

## ▪ 아쉬운 점

- 데이터셋의 수가 너무 작았다. (영화데이터 리뷰 감성분석 : 20만개, 인하대 데이터 : 1만개)  
=> 학습이 어느 정도로 잘 진행되었는지 평가하기 모호함
- 게시글의 좋아요와 싫어요로 감성분석을 하기에는 **좋아요 비중이 압도적으로 높았다.** (약 8대2 : 데이터 불균형)  
=> 과연 좋아요와 싫어요 수는 감성분석 라벨링으로 적절한가?

## ▪ 개선사항

- 인하광장, 인하뉴스, 자유게시판 뿐만 아니라 더 다양한 홈페이지의 글들을 수집하면 더 높은 성능을 기대할 수 있을 것이다.
- 전체 게시판의 조회수를 이용해 조회수 예측을 하면 조금 더 좋은 성능의 결과물이 나올 것 같다.
- Transformer의 변형 모델이나, BERT나 BigBird 같은 더 최신의 모델들도 적용해보면 좋을 것 같다.



# 목차

## Table of contents



NLP(Natural Language Processing)



Language Model (언어모델)



NLP의 활용 (감성분석 예제)

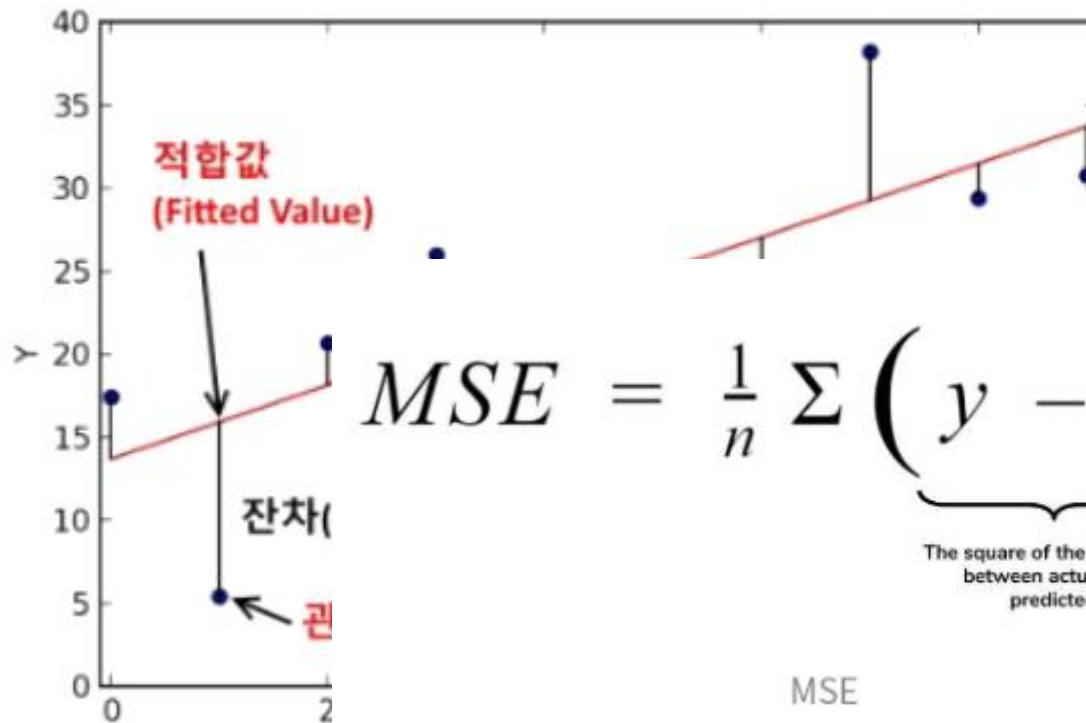


코드 설명



NLP와 통계학

# NLP와 통계학



➡ 2-2 회귀분석

$$MSE = \frac{1}{n} \sum \left( y - \hat{y} \right)^2$$

The square of the difference between actual and predicted

MSE

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \cdots + \beta_k x_{ki} + \varepsilon_i$$

다중 선형 회귀 모형

# NLP와 통계학

조건부확률의 연쇄법칙 (Chain rule for conditional probability)

$$P(x_1, x_2, x_3, \dots, x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2), \dots, P(x_n|x_1, \dots, x_{n-1})$$

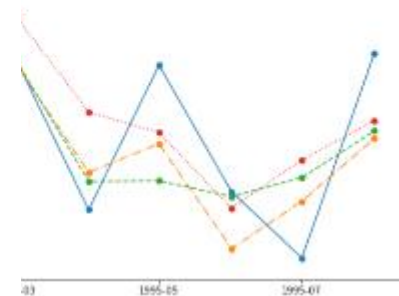
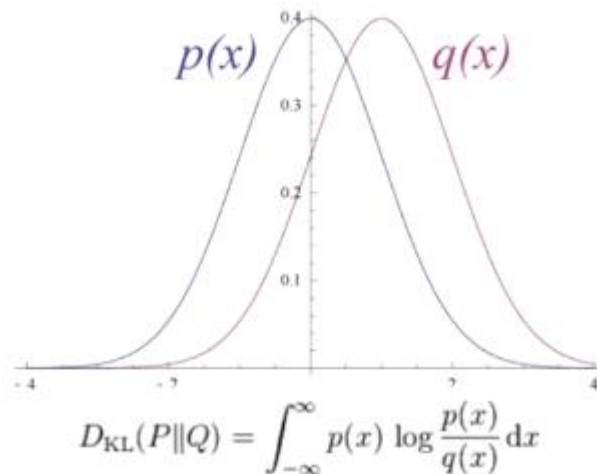
$$MSE = \frac{1}{N} \sum_i (pred_i - ta_i)^2$$



RMSE =

K-L Divergence

MAE =



$$P(B|A) = P(A) \times \frac{P(B|A)}{P(A)}$$

prior prior likelihood marginal

# Reference

- <https://arxiv.org/pdf/1706.03762.pdf>
  - [https://kazemnejad.com/blog/transformer\\_architecture\\_positional\\_encoding/#what-is-positional-encoding-and-why-do-we-need-it-in-the-first-place](https://kazemnejad.com/blog/transformer_architecture_positional_encoding/#what-is-positional-encoding-and-why-do-we-need-it-in-the-first-place)
  - <https://www.youtube.com/watch?v=AA621UofTUA>
  - <https://www.youtube.com/watch?v=h8avp8yDKV4&list=PLLENHvsRRLjDHllrXj0B8sz5-4xVbisBL&index=24>
  - <https://daeun-computer-uneasy.tistory.com/21>
  - <https://wikidocs.net/60691>
  - <https://ratsgo.github.io/blog/categories/>
  - <https://daeun-computer-uneasy.tistory.com/21>
  - <https://hyunsooworld.tistory.com/>
- 
- <https://github.com/gustn9609> => 모든 코드와 자료는 제 깃허브에 올려놓았습니다.



*Thank you.*