

# Chapter\_07\_요약\_question-answering\_v2.ipynb

[transformer](#) · 2024. 8. 16. 15:51

sequence classification

text classification

[0.8, 0.2] -> 확률분포

[31.2, 21.7] -> logit 하나의 위치(CLS) token에서만 logit vector를 뽑음

head

[representation vector, context vector]

[   ] [   ] [   ] [   ] [   ]

1 2 3 4 5

[CLS]   i       love       this       soup       .....

representation vector : 문장의 의미를 전부담고있는 vector

context vector : 문장의 의미를 전부담고있는 vector

[CLS] token을 맨앞에 두면 일관성있게 작업가능. CLS token 위치에 있는 vecor가 문장의 의미를 담고있음.

NER

token classification

[logits] [logits] [logits] [logits] [logits]

head   head   head   head -> 각단어에 head붙여서 각위치에서 logit vector를 뽑음

[ ] [ ] [ ] [ ] [ ]  
 1 2 3 4 5  
 [CLS] Pack love this store .....

QA (주어진 텍스트에서 찾을때만)

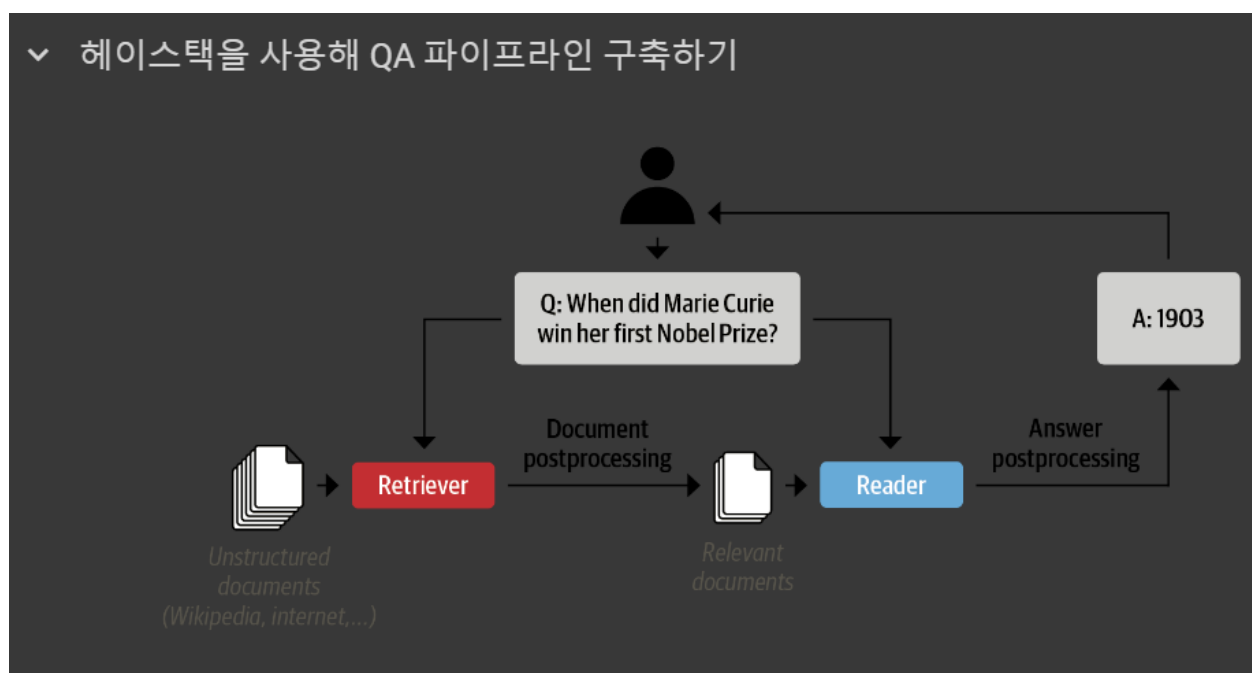
start and end logit -> 이렇게 해서 값이 2개씩나옴 -> 답변이 어디있는지 시작, 끝  
 $[v1, v2]$   $[v1, v2]$   $[v1, v2]$   $[v1, v2]$   $[v1, v2] \Rightarrow [v1, v1, v1, v1, v1], [v2, v2, v2, v2, v2]$   
 head head head head -> 5개모아서 이중 값이 크게 시작위치, 뒤쪽백  
 터는 가장큰값이 끝위치

[ ] [ ] [ ] [ ] [ ]  
 1 2 3 4 5  
 [CLS] Pack love this store .....

QA 파이프라인

해이스택은 QA파이프라인을 구축하기 편리하게 해주는 패키지

Retriever는 많은 문서중 관련 문서를 찾아서 reader(자연어모델)가 답을 찾아서 사용자에게 돌려준다.



(아래그림) 문서단어행렬

전체vocab을 열로 취하고, 문서를 row로 취해서 , 각 문서에 나온 단어갯수를 counting.

## 1. 문서 단어 행렬(Document-Term Matrix, DTM)의 표기법

문서 단어 행렬(Document-Term Matrix, DTM)이란 다수의 문서에서 등장하는 각 단어들의 빈도를 행렬로 표현한 것을 말한다. 쉽게 생각하면 각 문서에 대한 BoW를 하나의 행렬로 만든 것으로 생각할 수 있으며, BoW와 다른 표현 방법이 아니라 DTM 표현을 다수의 문서에 대해서 행렬로 표현하고 부르는 용어입니다. 예를 들어서 이렇게 4개의 문서가 있다고 합시다.

문서1 : 먹고 싶은 사과

문서2 : 먹고 싶은 바나나

문서3 : 길고 노란 바나나 바나나

문서4 : 저는 과일이 좋아요

띄어쓰기 단위 토큰화를 수행한다고 가정하고, 문서 단어 행렬로 표현하면 다음과 같습니다.

	과일이	길고	노란	먹고	바나나	사과	싶은	저는	좋아요
문서1	0	0	0	1	0	1	1	0	0
문서2	0	0	0	1	1	0	1	0	0
문서3	0	1	1	0	2	0	0	0	0
문서4	1	0	0	0	0	0	0	1	1

(아래그림)

$tf$  는 위 문서단어 행렬에서 나온 특정단어  $t$ 의 횟수

$idf$  는  $df$ 의 역수(반비례)

$df$ 가(특정단어)가 0일수도 있어서 분무가 0이 되면 문제되서 1을 더해준것

예를들어 the라는 특정단어인경우,

$df(t)$  가  $n$ 에 가까워지면서,

$\log(n/1+df)$ 는  $\log 1$ 이라서

0값이 된다.

**(1)  $tf(d,t)$  : 특정 문서  $d$ 에서의 특정 단어  $t$ 의 등장 횟수.**

생소한 글자때문에 어려워보일 수 있지만, 잘 생각해보면 TF는 이미 앞에서 구한 적이 있습니다. TF는 앞에서 배운 DTM의 예제에서 각 단어들이 가진 값들입니다. DTM이 각 문서에서의 각 단어의 등장 빈도를 나타내는 값이었기 때문입니다.

**(2)  $df(t)$  : 특정 단어  $t$ 가 등장한 문서의 수.**

여기서 특정 단어가 각 문서, 또는 문서들에서 몇 번 등장했는지는 관심가지지 않으며 오직 특정 단어  $t$ 가 등장한 문서의 수에만 관심을 가집니다. 앞서 배운 DTM에서 바나나는 문서2와 문서3에서 등장했습니다. 이 경우, 바나나의  $df$ 는 2입니다. 문서3에서 바나나가 두 번 등장했지만, 그것은 중요한 게 아닙니다. 심지어 바나나란 단어가 문서2에서 100번 등장했고, 문서3에서 200번 등장했다고 하더라도 바나나의  $df$ 는 2가 됩니다.

**(3)  $idf(t)$  :  $df(t)$ 에 반비례하는 수.**

$$idf(t) = \log\left(\frac{n}{1 + df(t)}\right)$$

$df(t)$ 가  $n$ 에 가까워지면서,  
 $\log(n/1+df(t))$ 는  $\log 1$ 이어서  
0값이 된다.

내 컴퓨터를 guys

**(1)  $tf(d,t)$  : 특정 문서  $d$ 에서의 특정 단어  $t$ 의 등장 횟수.**

생소한 글자때문에 어려워보일 수 있지만, 잘 생각해보면 TF는 이미 앞에서 구한 적이 있습니다. TF는 앞에서 배운 DTM의 예제에서 각 단어들이 가진 값들입니다. DTM이 각 문서에서의 각 단어의 등장 빈도를 나타내는 값이었기 때문입니다.

**(2)  $df(t)$  : 특정 단어  $t$ 가 등장한 문서의 수.**

여기서 특정 단어가 각 문서, 또는 문서들에서 몇 번 등장했는지는 관심가지지 않으며 오직 특정 단어  $t$ 가 등장한 문서의 수에만 관심을 가집니다. 앞서 배운 DTM에서 바나나는 문서2와 문서3에서 등장했습니다. 이 경우, 바나나의  $df$ 는 2입니다. 문서3에서 바나나가 두 번 등장했지만, 그것은 중요한 게 아닙니다. 심지어 바나나란 단어가 문서2에서 100번 등장했고, 문서3에서 200번 등장했다고 하더라도 바나나의  $df$ 는 2가 됩니다.

**(3)  $idf(t)$  :  $df(t)$ 에 반비례하는 수.**

$$idf(t) = \log\left(\frac{n}{1 + df(t)}\right)$$

바나나	$\ln(4/(2+1)) = 0.287682$
사과	$\ln(4/(1+1)) = 0.693147$
싫은	$\ln(4/(2+1)) = 0.287682$
저는	$\ln(4/(1+1)) = 0.693147$
좋아요	$\ln(4/(1+1)) = 0.693147$

문서3에서 길고노란 바나나 바나나  
의 바나나가 많이 나온 것에  $tf=2$ 이고,  
바나나가 전체문서중 2개문서에서 나와서  
 $idf$  가 가중치를 곱해서

문서의 총 수는 4이기 때문에  $\ln$  안에서 분자는 늘 4으로 동일합니다. 분모의 경우에는 각 단어가 등장한 문서의 수(DF)를 의미하  
는데, 예를 들어서 '먹고'의 경우에는 총 2개의 문서(문서1, 문서2)에 등장했기 때문에 2라는 값을 가집니다. 각 단어에 대해서 IDF  
의 값을 비교해보면 문서 1에만 등장한 단어와 문서2개에만 등장한 단어는 값의 차이를 보입니다. IDF는 여러 문서에서 등장한  
단어의 가중치를 낮추는 역할을 하기 때문입니다.

TF-IDF를 계산해보겠습니다. 각 단어의 TF는 DTM에서의 각 단어의 값과 같으므로, 앞서 사용한 DTM에서 단어 별로 위의 IDF값  
을 곱해주면 TF-IDF 값을 얻습니다.

	과일이	길고	노란	먹고	바나나	사과	싫은	저는	좋아요
문 서 1	0	0	0	0.287682	0	0.693147	0.287682	0	0
문 서 2	0	0	0	0.287682	0.287682	0	0.287682	0	0
문 서 3	0	0.693147	0.693147	0	0.575364	0	0	0	0
문 서	0.693147	0	0	0	0	0	0	0.693147	0.693147

아래그림)

출처:<https://wikidocs.net/31698>



## 04-04 TF-IDF(Term Frequency-I...

이번에는 DTM 내에 있는 각 단어에 대한 중요도를 계산할 수 있는 TF-IDF 가중치에 대해서 알아보겠습니다. TF-IDF를 ...

wikidocs.net

문서 하나의 vocab 에 있는 단어를 몇십에서 몇백개밖에 쓰지 않아서  
(TF, IDF 행렬을 쓰면 0을 갖는 단어는 항상0이라서 곱해서 0) sparse  
representation임

목적

문서를 vector화 하기 위한것.

vector하면 내적해서 similarity를 구할수 있기때문에 문서를 vector화 한다.

문제를 vector 화해서 유사도를 관련문서와(retriever) 유사도가 높은 것을 찾아내기위한것

-> Keyword (답높은 것찾는것)

을 곱해주면 TF-IDF 값을 얻습니다.

	과일이	길고	노란	먹고	바나나	사과	싫은	저는	좋아요
문 서 1	0	0	0	0.287682	0	0.693147	0.287682	0	0
문 서 2	0	0	0	0.287682	0.287682	0	0.287682	0	0
문 서 3	0	0.693147	0.693147	0	0.575364	0	0	0	0
문 서 4	0.693147	0	0	0	0	0	0	0.693147	0.693147

아래그림)

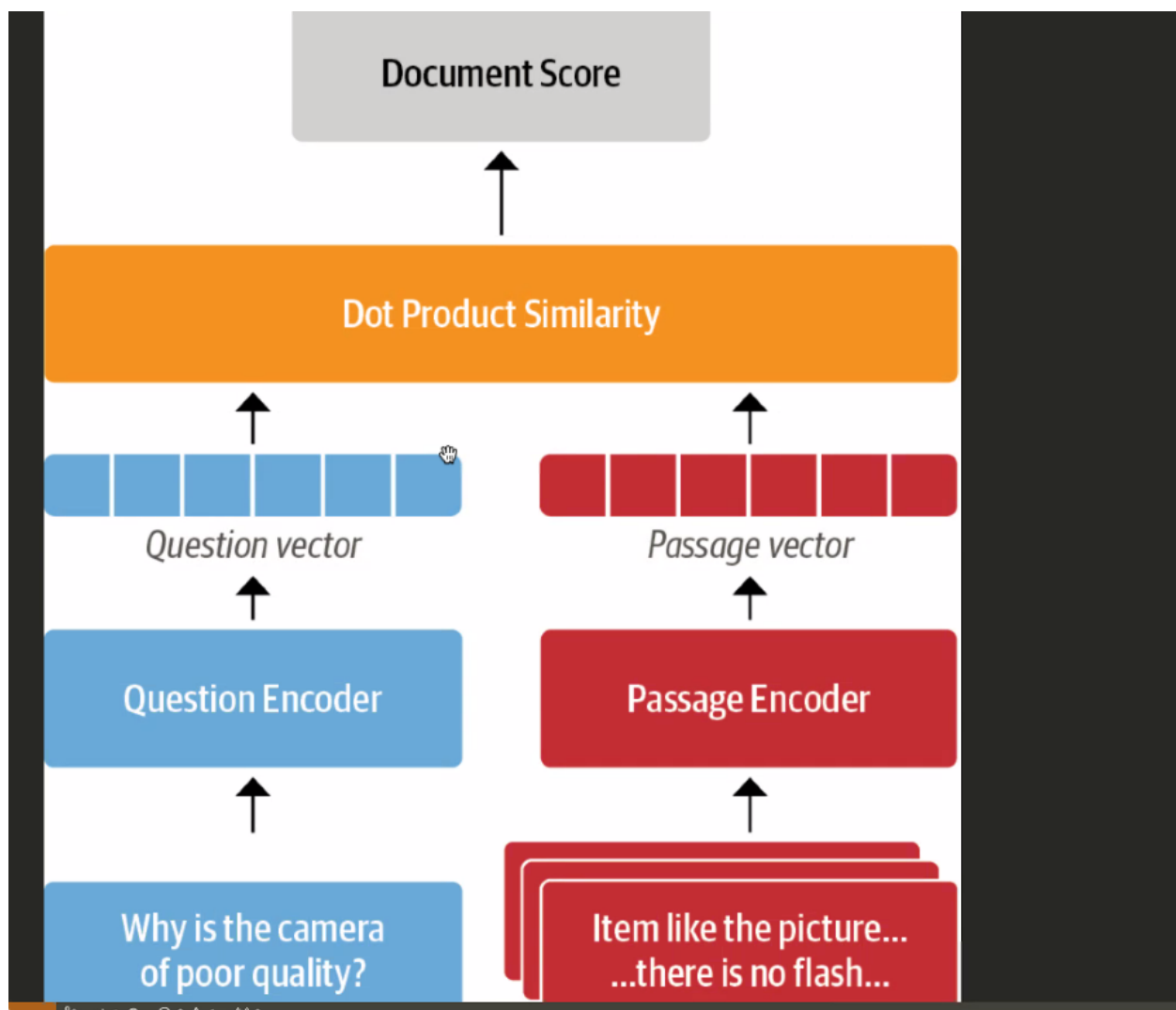
Dense retriever

sparse retriever는 자원이 많이드는 반면,

dense retriever는 반대임. 효율성이 높음. (

DPR

DENSE PASSAGE RETRIEVER



DPR

DENSE PASSAGE RETRIEVER

BM25(sparse retriever)DPR 와 별차이없음



## F1 score 소개

출처: <https://velog.io/@jadon/F1-score%EB%9E%80>

### F1 score란?

좋은 Article을 읽게 되어 헛갈리던 ML metrics에 대해 정리해보고자 한다. F1 score는 분류 모델에서 사용되는 머신러...

velog.io

출처: <https://images.app.goo.gl/1wXmAShHXcnZP6Yz6>



$$\begin{aligned} \text{F1 Score} &= \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} \\ &= \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \end{aligned}$$

$$\text{Accuracy} = \frac{\# \text{ of correct predictions}}{\# \text{ of total predictions}}$$

아래그림)

F1 Score : 성능지표 (precision과 recall의 조화평균)

pred: about 6000 hours

label: 6000 hours

precision = 모델이 예측한 답 중에 맞은 것의 비율  
= 2/3

recall = 실제 정답 중 모델이 찾은 것의 비율  
= 2/2 = 1

$$3/2 + 1/1 = 5/2$$

$$2 / (5/2) = 4/5 \Rightarrow 0.8$$

### 리더 평가하기

```

from haystack.modeling.evaluation.squad import compute_f1, compute_exact
# f1 = 2 / (1/precision + 1/recall)
pred = "about 6000 hours" # about, 6000, hours: 이 중 맞는 비율을 구한게 precision
label = "6000 hours"
print(f"EM: {compute_exact(label, pred)}") # Exact Match
print(f"F1: {compute_f1(label, pred)}")

[64] ✓ 0.0s
...
EM: 0
F1: 0.8

pred = "about 6000 dollars"
print(f"EM: {compute_exact(label, pred)}")
print(f"F1: {compute_f1(label, pred)}")

[65] ✓ 0.0s
...
EM: 0
F1: 0.4

from haystack.pipelines import Pipeline
def evaluate_reader(reader):

```

precision = 모델이 예측한 답 중에 맞은 것의 비율  
= 2/3

recall = 실제 정답 중 모델이 찾은 것의 비율  
= 2/2 = 1

$3/2 + 1/1 = 5/2$

$2 / (5/2) = 4/5 \Rightarrow 0.8$



공감



구독하기

#### 'transformer' 카테고리의 다른 글

<a href="#">09_few-to-no-labels.ipynb -text의 라벨을 tag하기위한 방법 (0)</a>	2024.08.30
<a href="#">08_model-compression.ipynb (0)</a>	2024.08.26
<a href="#">Chapter_06 요약_summarization.ipynb (0)</a>	2024.08.07
<a href="#">Chapter_05 요약_text-generation.ipynb (0)</a>	2024.07.31
<a href="#">Chapter_04 요약_multilingual-ner.ipynb (0)</a>	2024.07.26

관련글

관련글 더보기

예치금 납부 (입력자)	2023. 11. 10.(금) ~ 11. 14.(화) 16:00 마감 (본등록 의사 확인 절차/납부 시 학번 부여)	우리은행 전국지점 & 가상계좌
등록금 납부 (입력자)	2024. 1. 19.(금) ~ 1. 30.(화) 16:00 마감 ※ 본교 등록금 책정 절차에 따라 변경될 수 있음	우리은행 전국지점 & 가상계좌

### 09\_few-to-no-labels.ipynb -text의 라벨을 tag하기위한 방법

매뉴를 통해 원서접수 페이지(유웨이(영어/과외))로 이동해 정수할 수 있습니다.

(2) 필기 및 실기시험 시 (page)를 참조하시기 바랍니다.

(3) 위 지정일(10.13(금) / 우편소인 유효)까지 입학원서/학업계획서(유웨이(영어)출력물), 학위수여증명서, 성적증명서, 기타 증명서류를 지원 학과 사무실(24page~ 학과별 주소록 참고)로 우편 송부(제출하는 모든 서류는 원본이어야 함) 혹은 방문 접수하여야 하며, 미제출시 면접이 불가할 수 있습니다.

### 08\_model-compression.ipynb

Cannot export sdpa encoder to onnx when transformers(torch) > 4.43.0 (Occurred when translating scaled\_dot\_product\_attention)

LR = 0 when using DeepSpeed Config and LORA on Trainers

Whisper generate return a slice of result if result has more than one added tokens

resize\_token\_embeddings in NLLS leading to empty outputs

Support StaticCache in assisted generation

N-gram은 문장을 몇 개의 단어로 나누는지에 따라 분류가 결정됩니다. 아래와 같은 예시 문장을 활용하여 N-gram 언어 모델을 구현했을 때 어떤 결과가 나오는지 알아봄으로써 N-gram 종류에 대해 이해해 봅니다. 문장 부호는 전처리라고 가정하겠습니다.

"오늘 점심 메뉴는 뭐예요?"

### Chapter\_06 요약\_summarization.ipynb

$$y = y_1 y_2 \dots y_T$$

$$P(y_1 | x) = P(y_1 | y_1, x)$$

### Chapter\_05 요약\_text-generation.ipynb

2	Bigram(n=2)	오늘 점심, 점심 후전, 후전 메뉴, 메뉴 리스트, 리스트 피자
3	Trigram(n=3)	오늘 점심 후전, 점심 후전 메뉴, 후전 메뉴 리스트, 메뉴 리스트 피자
4	4-gram(n=4)	오늘 점심 후전 메뉴, 점심 후전 메뉴 리스트, 후전 메뉴 리스트 피자

$$= \prod_{t=r}^T P(y_t | y_{<t}, x^t)$$

# 자연어(NLP)

네이쳐2024 님의 블로그입니다.



구독하기 +

댓글 0



이름

비밀번호

내용을 입력하세요.



등록