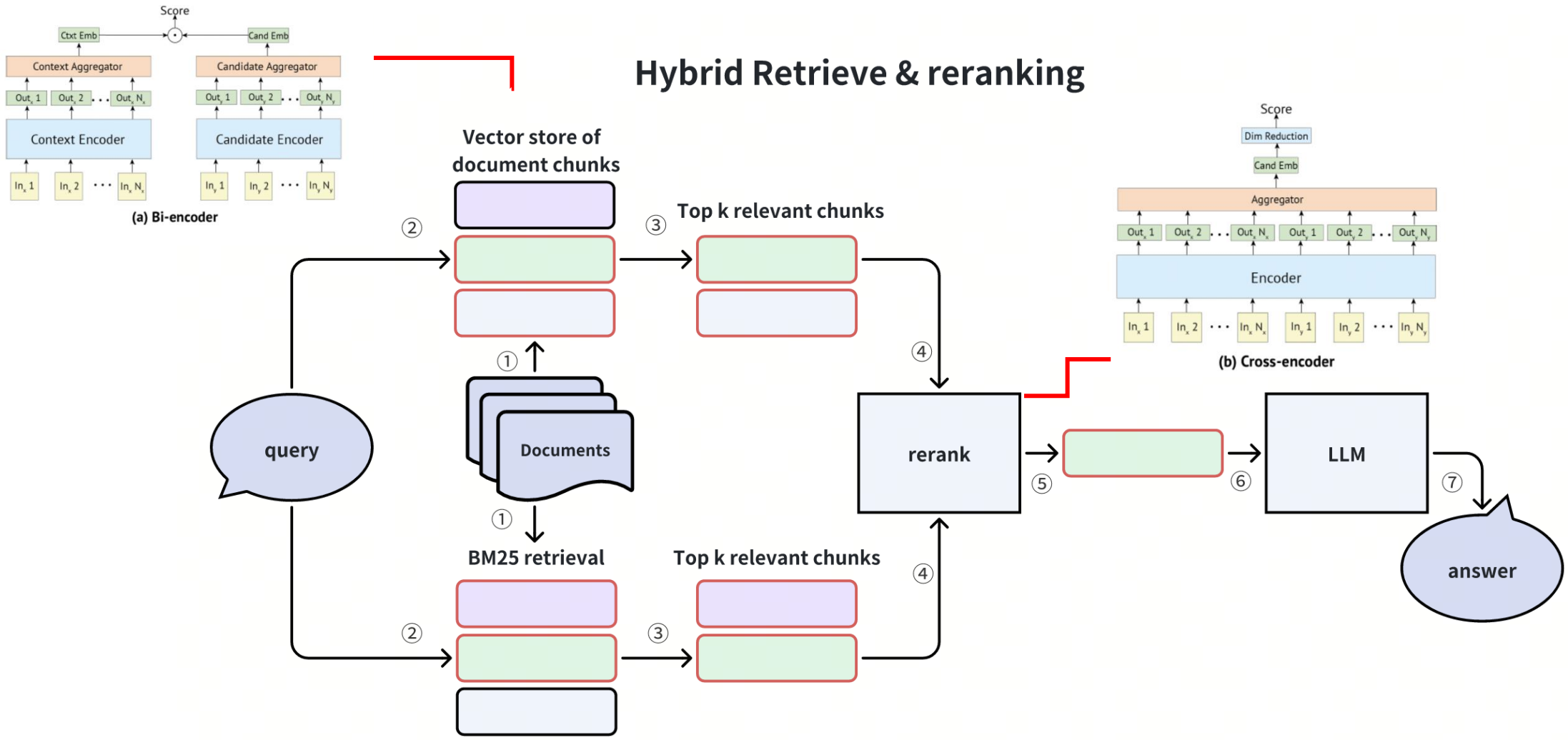
Modern Sparse Neural Retrieval

2025 동계세미나

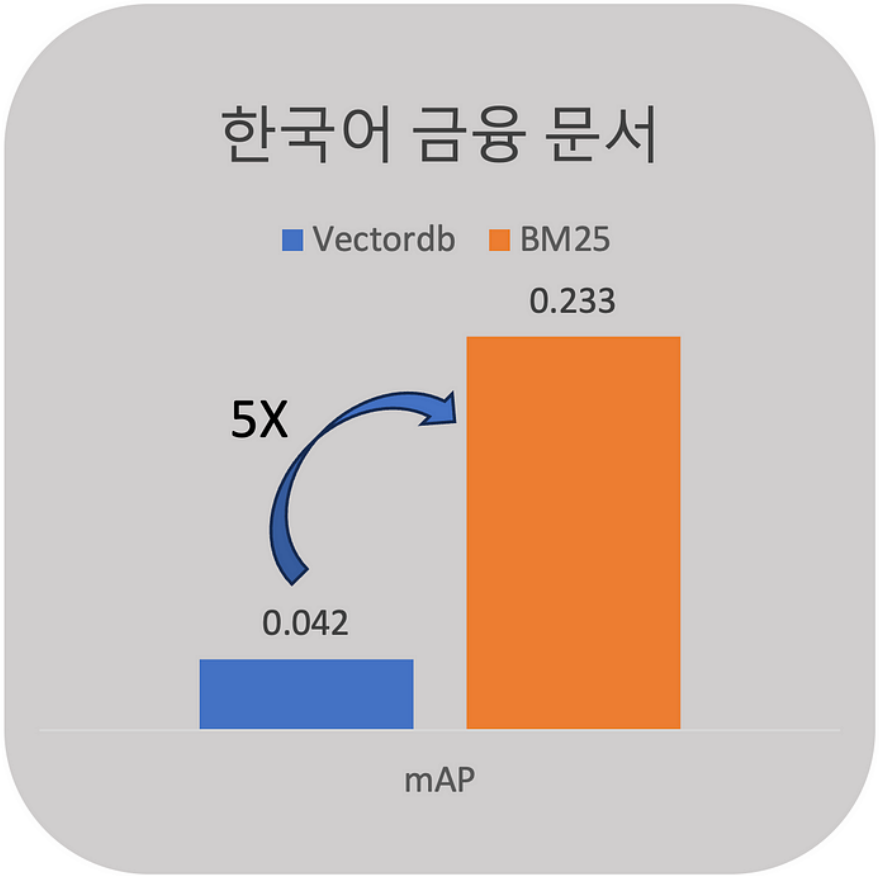
장영준

Table Of Contents

현대 Retrieval-Augmented Generation 패러다

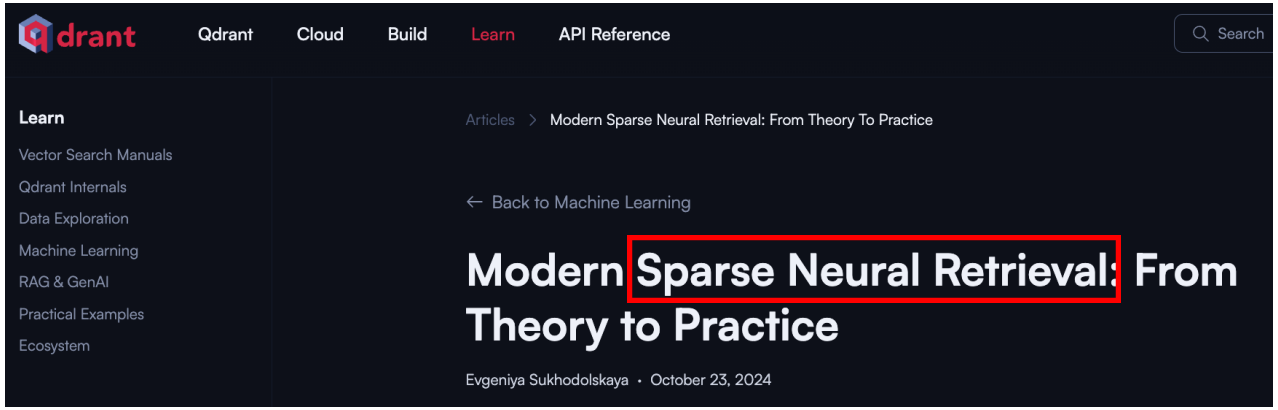


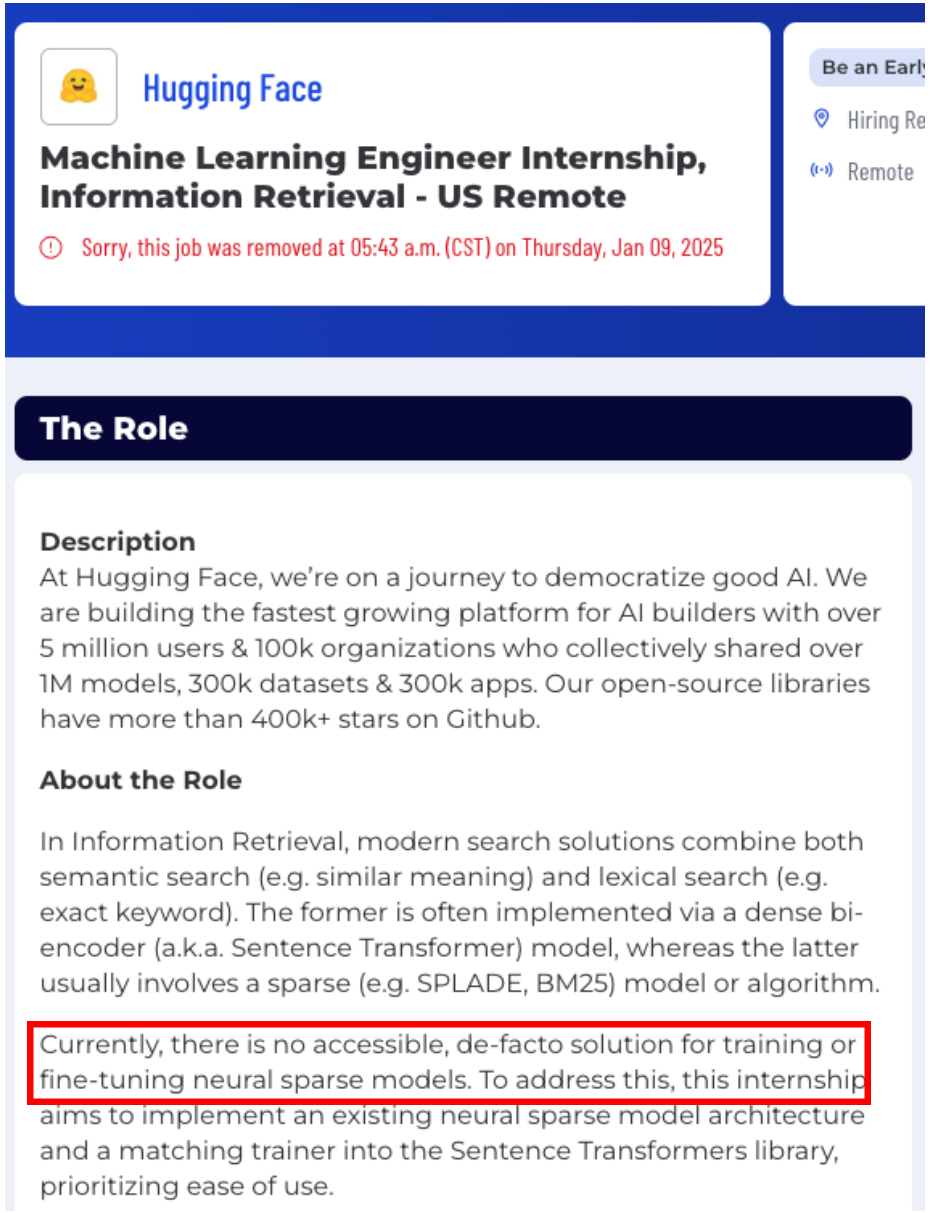
Sparse retrieval의 필요성



AutoRAG – "RAG에서 BM25가 더 좋을 수도 있습니다."

Sparse retrieval의 필요성





Sparse Neural Retrieval의 과거와 현재

0. Sparse vs. Dense vectors
1. TF-IDF, BM25
2. Doc2query
3. DeepCT
4. SparTerm
5. SPLADE
6. LENS

Sparse vs. Dense vectors

0. Sparse vs. Dense vectors

Difference?

0. Sparse vs. Dense vectors

Difference?

	SPARSE	DENSE
장점	1. 빠른 검색 가능 2. fine-tuning 필요 X 3. 검색된 경우, 설명 가능	1. 의미 반영 가능 2. 작은 벡터 안에 많은 정보 포함
단점	1. Vocab mismatch 2. 의미 반영 어려움	1. fine-tuning 필요 2. 느린 검색 시간 3. 해석 어려움

TF-IDF, BM25 and Background

1. TF-IDF, BM25, Backgrounds

TF-IDF

<TF-IDF 계산 방법>

$$TF(t, d) = \frac{\text{문서 d 에서 단어 t 가 등장한 횟수}}{\text{문서 d 에 등장한 모든 단어의 수}}$$

$$IDF(t, D) = \log\left(\frac{\text{총 문서의 개수}}{\text{단어 t 를 포함하는 문서의 수}}\right)$$

$$TF\text{-}IDF(t,d,D) = TF(t, d) * IDF(t, D)$$

특정 문서에서는 많이 등장하지만,
총 문서에서는 많이 등장하지 않는 단어가 높은 스코어 받음

- "이, 그, 저" 등의 대명사 -> 낮은 스코어
- "오스트랄로피테쿠스" -> 높은 스코어

<TF-IDF의 문제점>

- 문서가 길어질수록, 단어 빈도가 높아질 가능성이 커짐
  (긴 문서에서 TF-IDF 점수가 커질 확률이 높음)
- 단어가 문서 내에서 반복적으로 나오면 점수가 선형적으로 증가
  (e.g. 아파트아파트 아파트아파트)

1. TF-IDF, BM25, Backgrounds

BM25

$$\text{score}(D,Q) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i,D) \cdot (k_1+1)}{f(q_i,D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\text{avgdl}}\right)}$$

- f(qi, D): TF (t 등장 횟수/d의 모든 단어
- k1, b: 하이퍼파라미터
- avgdl: 문서 평균 길이

- f(qi, D)=10과 f(qi, D)=1000 비교
  - TF: 100배 차이
  - BM25:
    - 10*(1.5) / (10+0.5) = 1.43
    - 1000*(1.5) / (1000 + 0.5) = 1.5
    - 약 1.x 배 차이

- avgdl로 길이에 대한 패널티
  - D > avgdl -> 분모 커짐 -> 전체 스코어 작아짐
  - D < avgdl -> 분모 작아짐 -> 전체 스코어 커짐

1. TF-IDF, BM25, Backgrounds

BM25

$$\text{score}(D,Q) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\text{avgdl}}\right)}$$

- f(qi, D): TF (t 등장 횟수/d의 모든 단어
- k1, b: 하이퍼파라미터
- avgdl: 문서 평균 길이

$$\text{IDF}(q_i) = \ln\left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1\right)$$

- N: 전체 문서 수
- N(qi): qi가 나타나는 문서 수

- TF-IDF에 비해 자주 등장하는 단어는 가중치 낮게, 드물게 등장하는 단어들은 가중치 높게 !
- n(qi)가 작을수록 (특정 단어가 적은 문서에서 등장할수록), 분모는 작아지고 분자는 커지므로 IDF 값은 커진다.
  반대로 n(qi)가 클수록 분모는 커지고 분자가 작아지므로 IDF 값이 작아진다.

1. TF-IDF, BM25, Backgrounds

검색 전 색인 (Indexing)

효율적인 검색 위해 역색인 (Inverted Index) 사용

문서 ID	내용
doc1	"apple banana cherry"
doc2	"banana cherry date"
doc3	"cherry date apple"

1. Forward Index (일반 색인)

- doc1: {"apple": 1, "banana":1, "cherry":1}
- doc2: {"banana":1, "cherry":1, "date": 1}
- doc3: {"cherry":1, "date": 1, "apple":1}

2. Inverted Index (역색인)

- apple: {"doc1": 1, "doc3": 1}
- banana: {"doc1":1 , "doc2": 1}
- cherry: {"doc1":1 , "doc2": 1, "doc3":1}
- date: {"doc2": 1, "doc3":1}

1. TF-IDF, BM25, Backgrounds

Metric

- MRR

$$MRR@10 = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{\text{Rank}_i}, \quad \text{Rank}_i \leq 10$$

Query	Proposed Results (Top 10)	Correct Response	Rank	Reciprocal Rank
cat	catten, cati, cats	cats	3	$\frac{1}{3}$
torus	torii, tori, toruses	tori	2	$\frac{1}{2}$
virus	unrelated_1, unrelated_2, ...	viruses	-	0

$$MRR@10 = \frac{1}{3}\left(\frac{1}{3}+\frac{1}{2}+0\right) = \frac{5}{18} \approx 0.278$$

1. TF-IDF, BM25, Backgrounds

Metric

- MAP

$$MAP = \frac{1}{Q}\sum_{i=1}^{Q} AP_i$$

Rank	Rel/Unrel	Precision	Recall
1	Rel	1/1	1/3
2	Unrel	1/2	1/3
3	Rel	2/3	2/3
4	Unrel	2/4	2/3
5	Rel	3/5	3/3

$$AP = \frac{1 + \frac{2}{3} + \frac{3}{5}}{3} = 0.733$$

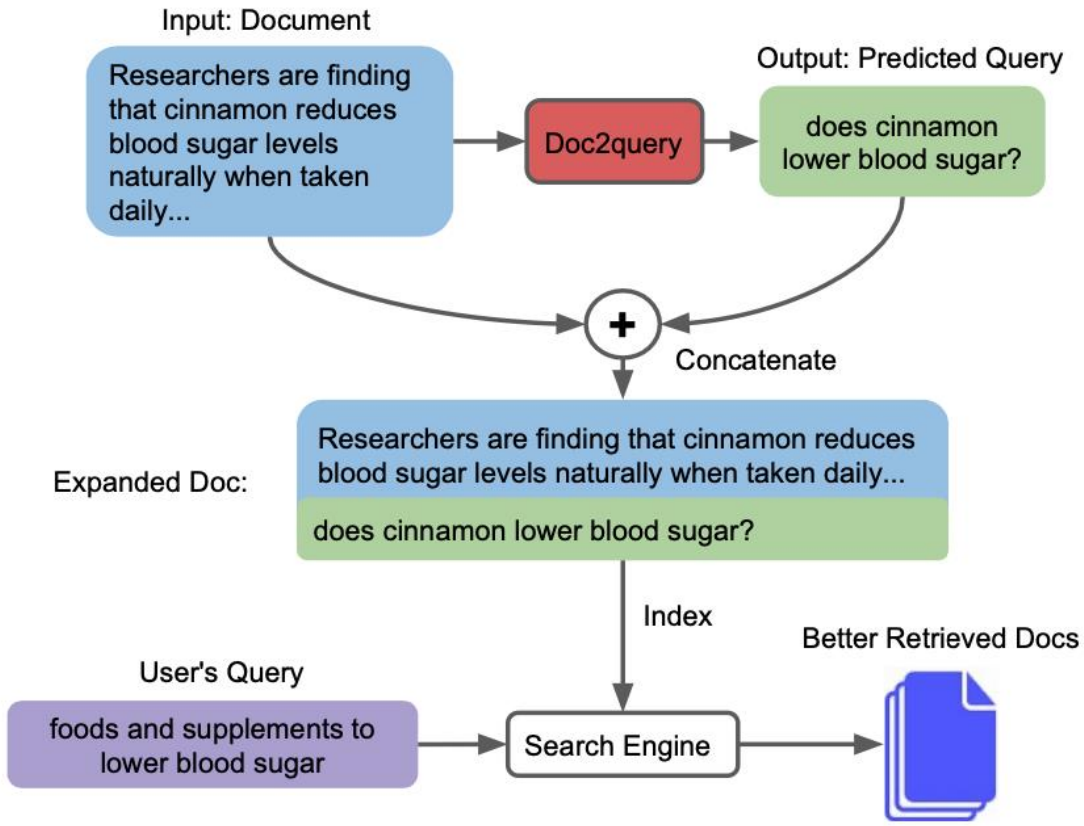
Document Expansion by Query Prediction

Rodrigo Nogueira,[1] **Wei Yang,**[2] **Jimmy Lin,**[2] and **Kyunghyun Cho**[3,4,5,6]

[1] Tandon School of Engineering, New York University
[2] David R. Cheriton School of Computer Science, University of Waterloo
[3] Courant Institute of Mathematical Sciences, New York University
[4] Center for Data Science, New York University
[5] Facebook AI Research [6] CIFAR Azrieli Global Scholar

Doc2query

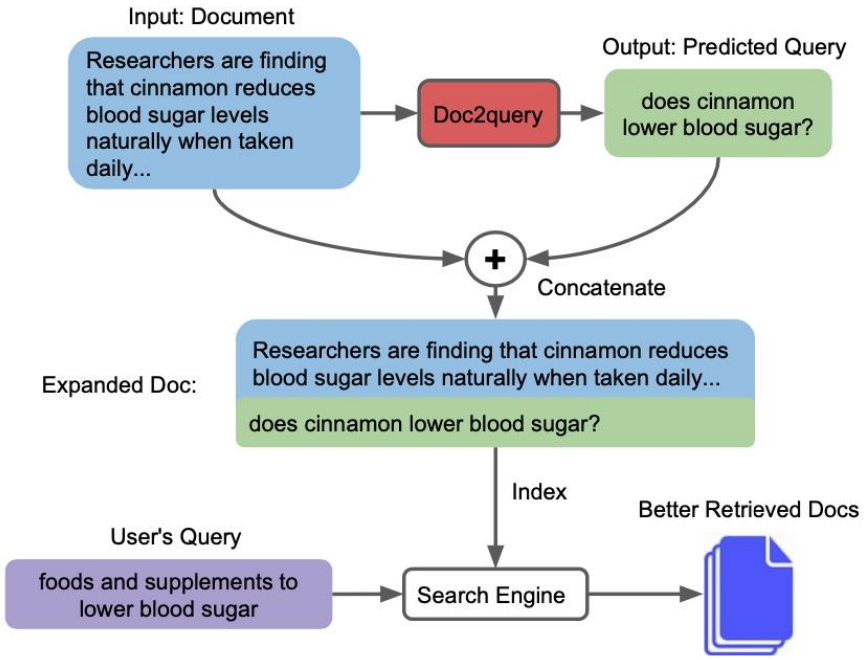
Problem?



- TF-IDF, BM25의 치명적인 단점은 "Keyword mismatch" 문제이다.
- 따라서, 생성형 모델을 통해 Expansion하여 이 문제를 해결하자 ! (Document expansion)

Doc2query

Framework



1. Passage-query 데이터 준비
2. Transformer (from scratch)로 passage가 주어졌을 때 query를 생성하도록 학습
3. 학습 완료 후, eval dataset의 document로 top-k random sampling 기법을 통해 10개의 query 만들어둠
4. 만들어진 queries를 document에 더함 (그냥 string 끼리 concatenate)
5. Indexing 하여 검색에 사용

Doc2query

Evaluation

		TREC-CAR MAP	MS MARCO MRR@10		Retrieval Time ms/query
		Test	Test	Dev	
Reranker	Single Duet v2 (Mitra and Craswell, 2019b)	-	24.5	24.3	650*
	Co-PACRR♠ (MacAvaney et al., 2017)	14.8	-	-	-
Single-stage retrieval	BM25	15.3	18.6	18.4	50
	BM25 + RM3	12.7	-	16.7	250
	BM25 + Doc2query (Ours)	18.3	21.8	21.5	90
	BM25 + Doc2query + RM3 (Ours)	15.5	-	20.0	350
Multi-stage retrieval	BM25 + BERT (Nogueira and Cho, 2019)	34.8	35.9	36.5	3400†
	BM25 + Doc2query + BERT (Ours)	**36.5**	**36.8**	**37.5**	3500†

- Reranker는 당시 SOTA reranker
- 2019년에는 Sparse retriever로 1차 retrieval, BERT로 2차 reranking을 수행했음
- RM3는 query expansion 기법

Doc2query

Qualitative Analysis

Input Document:	July is the hottest month in Washington DC with an average temperature of 27C (80F) and the coldest is January at 4C (38F) with the most daily sunshine hours at 9 in July. The wettest month is May with an average of 100mm of rain.
Predicted Query:	weather in washington dc
Target query:	what is the temperature in washington
Input Document:	The Delaware River flows through Philadelphia into the Delaware Bay. It flows through and aqueduct in the Roundout Reservoir and then flows through Philadelphia and New Jersey before emptying into the Delaware Bay.
Predicted Query:	what river flows through delaware
Target Query:	where does the delaware river start and end
Input Document:	sex chromosome - (genetics) a chromosome that determines the sex of an individual; mammals normally have two sex chromosomes chromosome - a threadlike strand of DNA in the cell nucleus that carries the genes in a linear order; humans have 22 chromosome pairs plus two sex chromosomes.
Predicted Query:	what is the relationship between genes and chromosomes
Target Query:	which chromosome controls sex characteristics

Table 2: Examples of query predictions on MS MARCO compared to real user queries.

- Document에 있는 단어들이 query로 생성 -> term re-weighting
- Document에 없는 단어들이 query로 생성 -> document mismatch problem 완화
- (생성된 단어 : 새로운 단어) 비율 -> 31 : 69

Context-Aware Sentence/Passage Term Importance Estimation For First Stage Retrieval

Zhuyun Dai
Carnegie Mellon University
zhuyund@cs.cmu.edu

Jamie Callan
Carnegie Mellon University
callan@cs.cmu.edu

DeepCT

Problem?

In some cases, an upset **stomach** is the result of an allergic reaction to a certain type of food. It also may be caused by an irritation. Sometimes this happens from consuming too much alcohol or caffeine. Eating too many fatty foods or too much food in general may also cause an upset **stomach**.

All parts of the body (muscles , brain, heart, and liver) need energy to work. This energy comes from the food we eat. Our bodies digest the food we eat by mixing it with fluids(acids and enzymes) in the **stomach**. When the **stomach** digests food, the carbohydrate (sugars and starches) in the food breaks down into another type of sugar, called glucose.

- BM25와 doc2query 방법론들은 문맥적 의미에 대한 반영이 부족하다.
- 많은 query들이 document의 key idea에 기반하여 나오기 때문에 (가정),
  어떤 단어가 문장의 central한 의미를 반영하는지 아는 것이 중요하다 !!

DeepCT

Framework

- 문맥으로부터 각 단어의 importance score를 예측하기 위해 BERT 사용

$$\hat{y}_{t,c} = \vec{w}T_{t,c} + b$$

- T_{t,c}: text c 안에서의 term t의 embedding
- w&b: linear combination
- y^hat: 예측값

- GT Importance score 정의
  - passage d와 유관한 queries (여러 개) 중 term t를 포함하고 있는 query의 비율

$$QTR(t,d) = \frac{|Q_{d,t}|}{|Q_d|}.$$

- MSE Loss 계산

$$loss_{MSE} = \sum_c \sum_t (y_{t,c} - \hat{y}_{t,c})^2.$$

DeepCT

검색 전 Document 색인

학습된 DeepCT 모델에 텍스트 입력하면, 각 term의 importance score 출력

1. 모든 passage에 대해 DeepCT 모델로 inference한 후, predicted weights scaling (N=100)

$$\mathrm{TF}_{\mathrm{DeepCT}}(t,d) = round(\hat{y}_{t,d} * N),$$

2. Integer scale의 TF_DeepCT를 기존 BM25의 TF 자리에 배치

$$\mathrm{score}(D,Q) = \sum_{i=1}^{n} \mathrm{IDF}(q_i) \cdot \frac{f(q_i,D) \cdot (k_1+1)}{f(q_i,D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\mathrm{avgdl}}\right)},$$

3. TF 대신에 TF_DeepCT indexing

- 기존의 inverted index: “DNA”: {“doc_1”: 3, “doc_5” : 2}
- DeepCT inverted index: “DNA”: {“doc_1”: 9, “doc_5”: 7}
  (이렇게 importance score가 색인됨)

Experiment

- Dataset
  - MSMARCO[train] for train, MSMARCO[dev] for eval
  - TREC-CAR[train] for train, TREC-CAR[valid] for eval
- Baselines
  - TF index
  - TextRank
  - Doc2query
  - DeepCT-Index
  - DeepCT_W-Index: DeepCT에서 BERT를 context-independent embeddings로 바꾼 버전
    - 이 당시에는 word2vec 모델로 각 word embedding에서 passage embedding을 빼서 "문맥에 독립적인 단어 임베딩"을 구함
  - DeepCT_E-Index: DeepCT에서 BERT를 ELMo로 바꾼 버전

DeepCT

Results

Index	MS MARCO dev				TREC-CAR					
	BM25		QL		BM25			QL		
	MRR@10	W/T/L	MRR@10	W/T/L	MRR@10	MAP	W/T/L	MRR@10	MAP	W/T/L
tf index	0.191	-/-/-	0.189	-/-/-	0.233	0.174	-/-/-	0.211	0.162	-/-/-
TextRank	0.130	662/4556/1762	0.134	702/4551/1727	0.160	0.120	167/1252/441	0.157	0.118	166/1327/367
Doc2Query	0.221*	1523/4431/1026	0.224*	1603/4420/957	-	0.178	-/-/-	-	-	-/-/-
DeepCT-Index	**0.243***†	2022/3861/1097	**0.230***	1843/4027/1110	**0.332***	**0.246***	615/1035/210	**0.330***	**0.247***	645/1071/144
DeepCT$_W$-Index	0.174	931/4804/1245	0.168	867/4793/1320	0.205	0.147	250/1311/309	0.192	0.139	245/1345/280
DeepCT$_E$-Index	0.234*†	1891/4139/950	0.220*	1726/4210/1044	0.280*	0.201*	516/1144/200	0.276*	0.197*	540/1190/130

- 기존 SOTA였던 doc2query와 같은 데이터로 학습되었지만, DeepCT-Index가 가장 좋았다.
- DeepCT_W-Index, DeepCT_E-Index 모두 BERT 기반보다 성능이 낮다 — BERT model의 중요성을 강조

DeepCT

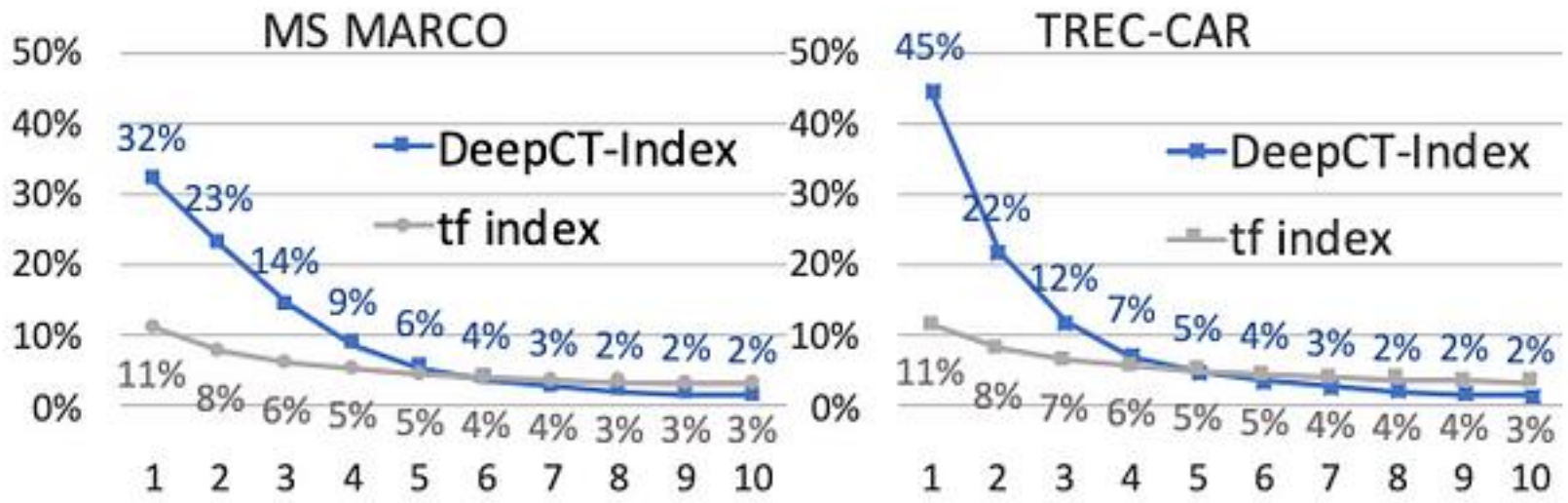
Qualitative Analysis

	Percentage of weights a term takes in the passage: 0 10% 20% 30% 40% >50%
Query	**who is susan boyle**
On-Topic	Amateur vocalist **Susan Boyle** became an overnight sensation after appearing on the first round of 2009's popular U.K. reality show Britain's Got Talent.
Off-Topic	Best Answer: a troll is generally someone who tries to get attention by posting things everyone will disagree, like going to a **susan boyle** fan page and writing **susan boyle** is ugly on the wall. they are usually 14-16 year olds who crave attention.

- 실제로 "Susan Boyle" 에 대해 물어보는 query에 대해,
  - 유관한 document (실제로 Susan Boyle에 대해 설명하는 document)에서는 해당 term의 weight가 높게,
  - 무관한 document (예시로 Susan Boyle이 등장하는 document)에서는 해당 term의 weight가 낮게 측정

DeepCT

Qualitative Analysis



- 각 document의 top10 highest-weighted term 분포
- 분포가 flat한 original tf index에 비해, DeepCT-Index는 특정 단어에 weight가 크게 걸림
- 따라서 DeepCT-Index가 적은 중심 단어들은 강조하고, 다른 단어들은 억제

DeepCT

느낀점

- 꼭 query가 document의 key point에서 나온다고 가정하는 것이 좋은 건지 모르겠다.
  (e.g. 수능 문제)
- Doc2query는 비효율적이더라도 term expansion을 통해 "term mismatch" 문제를 해결했는데, DeepCT는 그러지 못한다.

SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval

Yang Bai*†
Tsinghua University

Xiaoguang Li*
Huawei Noah's Ark Lab

Gang Wang
Huawei Noah's Ark Lab

SparTerm

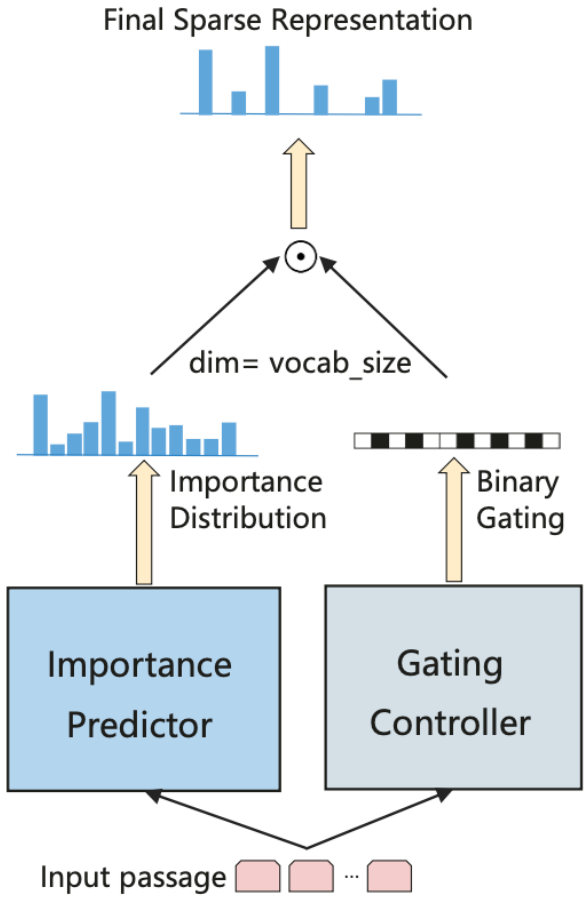
Problem?

Query	Can hives be a sign of pregnancy?	
Type	Term frequency	SparTerm
Literal term Weights	hives are caused by allergic reactions . the dryness and stretching of your skin along with other changes can make you more susceptible to experiencing hives during pregnancy . hives can be caused by an allergic reaction to almost anything . some common causes of hives during pregnancy are noted below : medicine	hives are caused by allergic reactions . the dryness and stretching of your skin along with other changes can make you more susceptible to experiencing hives during pregnancy . hives can be caused by an allergic reaction to almost anything . some common causes of hives during pregnancy are noted below : medicine
Term expansion		symptoms:1.0, women:0.99, rash:0.98, feel:0.99, causing:0.97, body:0.96, affect:0.96, baby:0.94, pregnant:0.93, **sign:0.91**, …

- BM25와 doc2query 방법론들은 문맥적 의미에 대한 반영이 부족하다.
- DeepCT는 문맥적 의미를 잘 반영하지만, "term mismatch" 문제를 해결하지 못한다.
- 따라서, SparTerm에서는 semantic하게 term을 확장한다.

SparTerm

Architecture

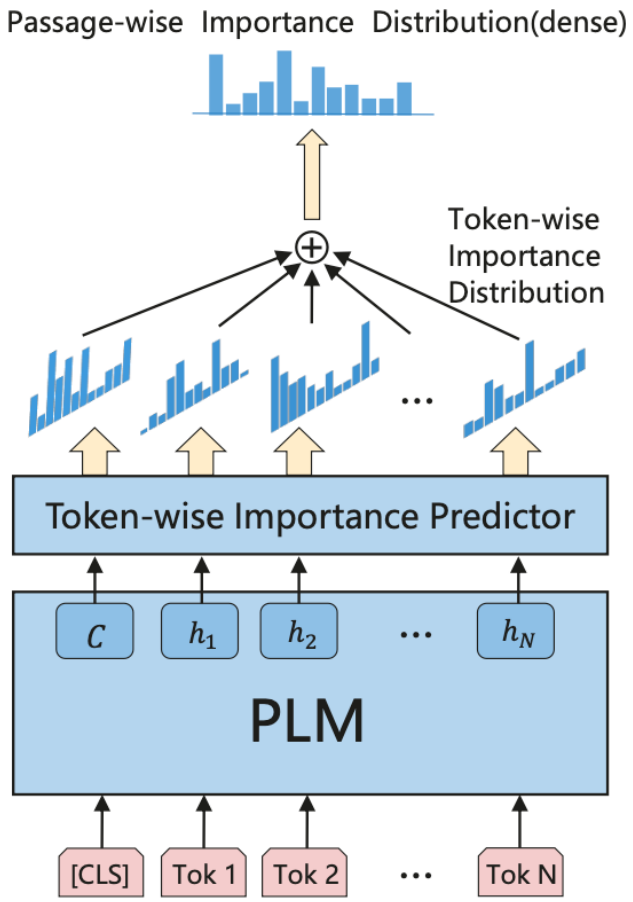


(a) SparTerm Model

$$p' = \mathcal{F}(p) \odot \mathcal{G}(p)$$

SparTerm

Importance Predictor



(b) Importance Predictor

1. Input passage p의 각 term에 대해 vocab으로부터의 semantic importance 출력

$$I_i = Transform(h_i)E^T + b$$

- 위의 식은 BERT의 MLM 식과 같음.

  h_i 는 i번째 token이 BERT를 통과해서 나오는 hidden state임.

2. 이후 통과하는 Token-wise Importance Predictor는 BERT의 MLM head와 같은 역할
3. 최종 output은 각 token 자리에서 BERT의 모든 vocab에 대한 distribution

따라서 왼쪽 그림의 PLM 부분을 BERT로 initialize. 이후 각 token 자리에서의 모든 distribution의 총합을 구함 $I = \sum_{i=0}^{L} Relu(I_i)$

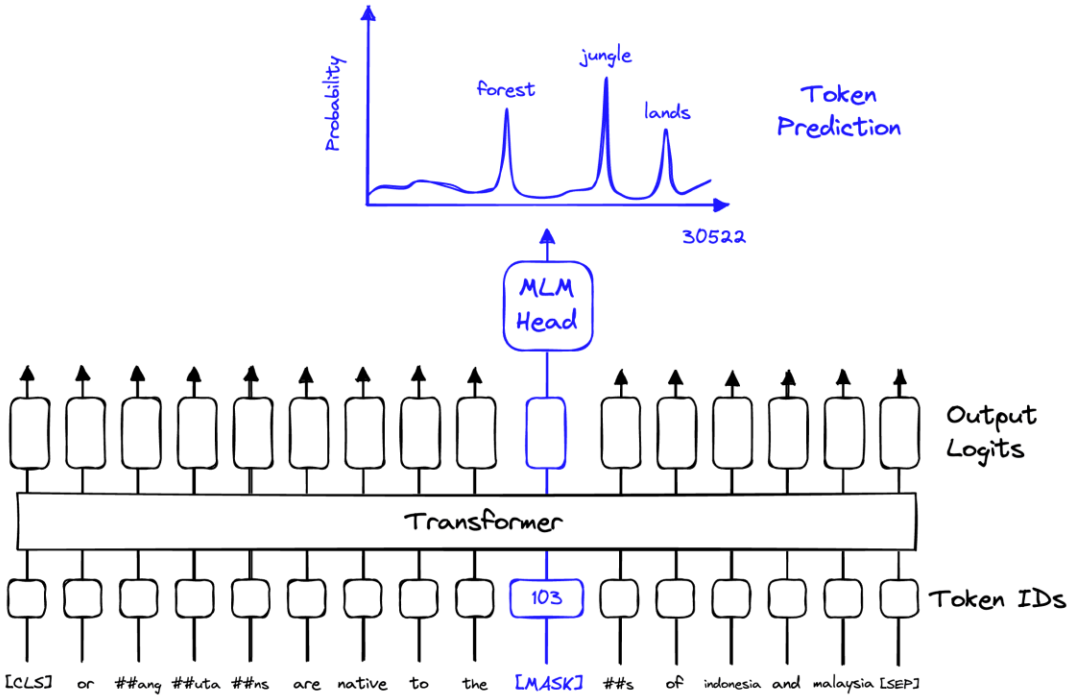
요약: 모든 문맥을 반영한 BERT vocab 크기의 distribution

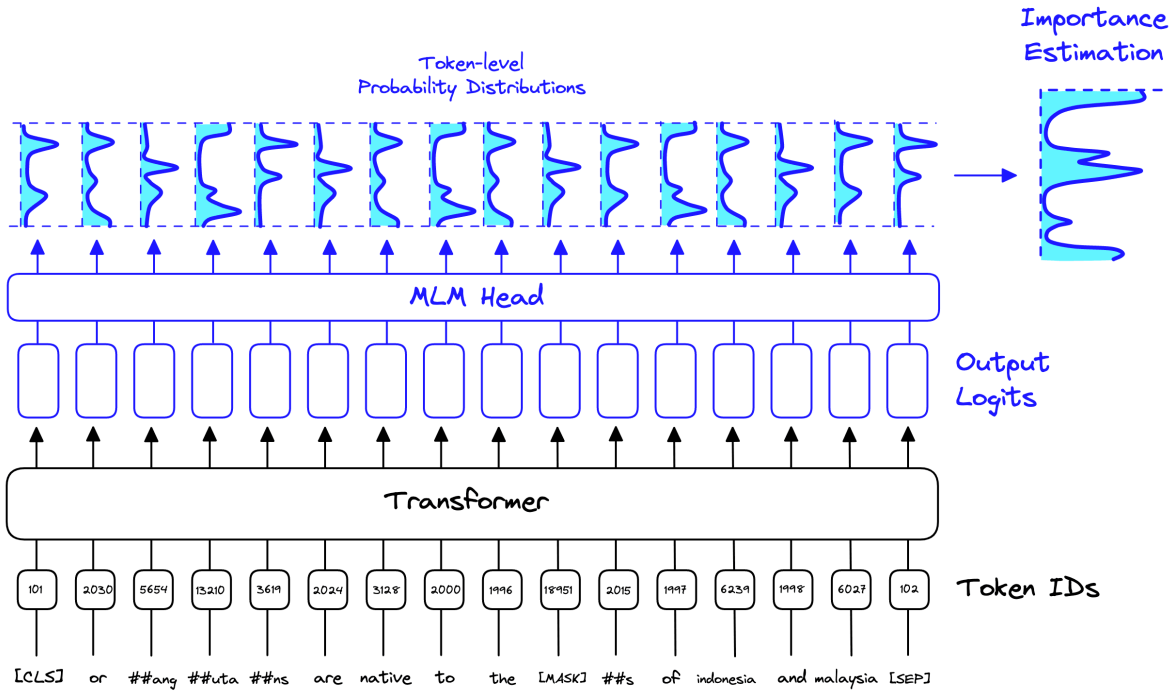
SparTerm

Importance Predictor

"orangutans are native to the rainforests of Indonesia and Malaysia"

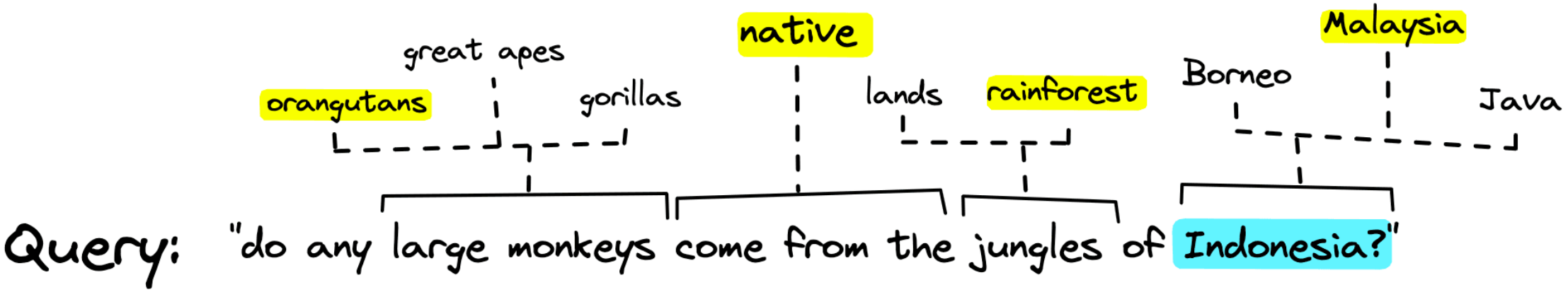
"orangutans are native to the [MASK] of Indonesia and Malaysia"





SparTerm

Importance Predictor



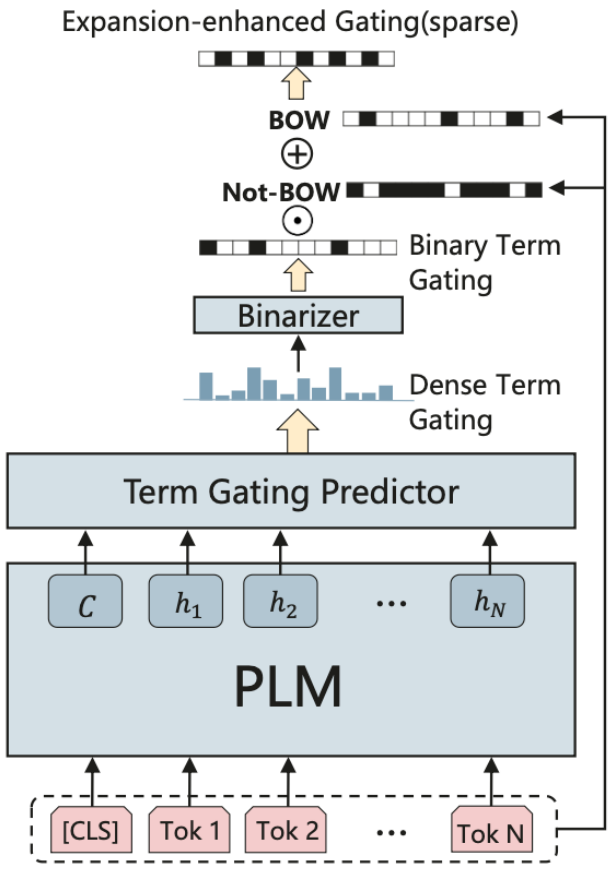
with query expansion

without query expansion

Doc: "Orangutans are native to the rainforests of Indonesia and Malaysia"

SparTerm

Gating Controller



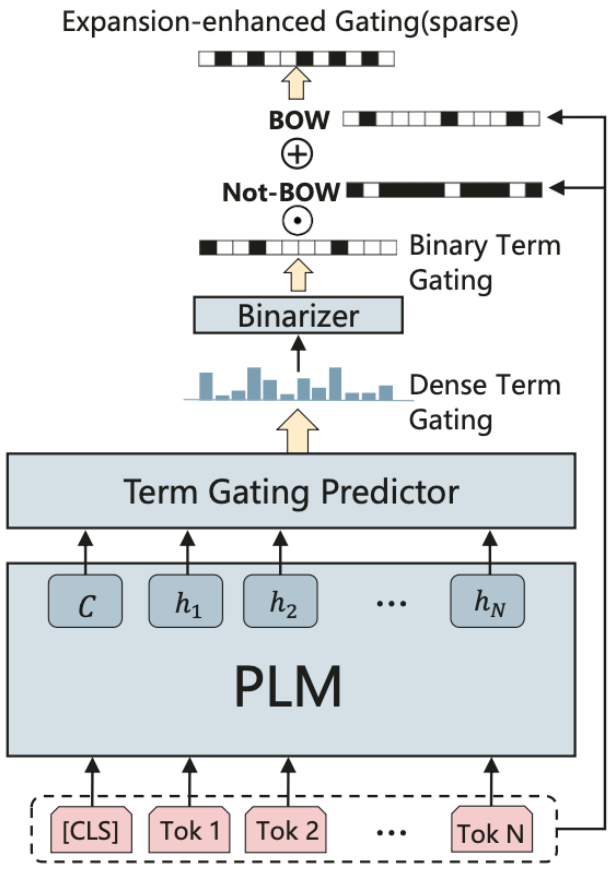
(c) Gating Controller

Passage에서 너무 확장되면 안되니까, passage를 표현하기 위해
어떤 term을 activate 시켜야 하는지 결정하는 binary signal을 Gating Controller에서 결정

1. 원래 주어진 passage에 있는 term들은 모두 activate
   -> **Literal-only gating**
2. Lexical mismatch 해결을 위해, passage topic과 관련한 term도 추가적으로 activate
   -> **Expansion-enhanced gating**

SparTerm

Gating Controller



(c) Gating Controller

1. **Literal-Only Gating**: 원본 문장에 나타나는 단어만 activate

$$G(p) = BoW(p)$$

2. **Expansion-enhanced Gating**: 다른 term 포함하여 activate

2-1. Importance Predictor와 같이, passage의 dense term 확률 분포 G 얻음

2-2. 모든 vocab 중 logit 값이 k(=0.7) 이상인 단어들만 G′에 저장

$$G' = Binarizer(G)$$

2-3. 높은 logit 값을 가지는, 원본 passage에는 없는 token 남김

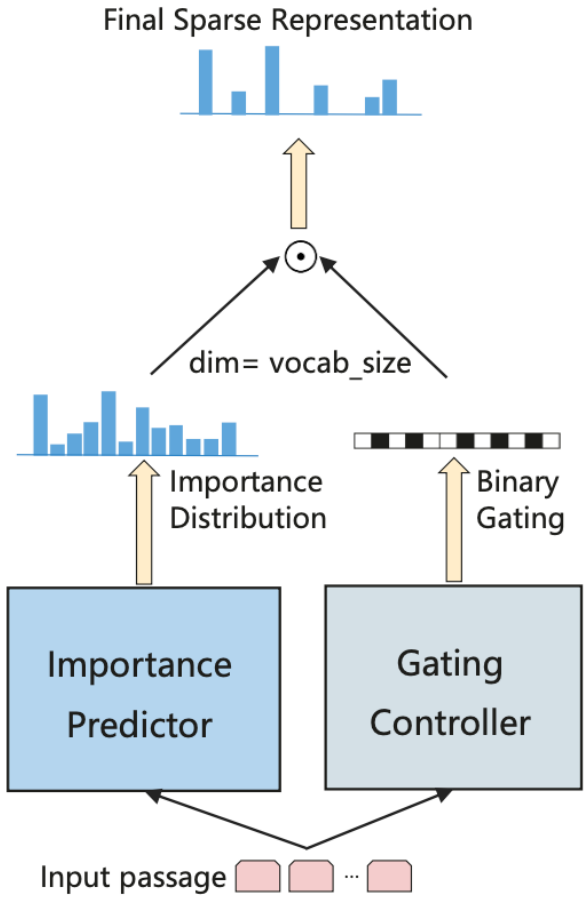
$$G_e = G' \odot \neg \text{BoW(p)}$$

2-4. 최종 expansion-enhanced gating vector

$$G_{le} = G_e + BoW(p)$$

SparTerm

Review



(a) SparTerm Model

$$p' = \mathcal{F}(p) \odot \mathcal{G}(p)$$

SparTerm

Training

Importance Predictor – Triplet Loss

$$L_{rank}(q_i, p_{i,+}, p_{i,-}) = -\log \frac{e^{sim(q_i', p_{i,+}')}}{e^{sim(q_i', p_{i,+}')} + e^{sim(q_i', p_{i,-}')}}$$

SparTerm

Training

Term expansion	Description and examples
Passage2query	Expand words that tend to appear in corresponding queries, i.e. “how far”, “what causes”.
Synonym	Expand synonym for original core words, i.e. “cartoon”->“animation”.
Co-occurred words	Expand frequently co-occurred words for original core words, i.e. “earthquakes”->“ruins”.
Summarization words	Expand summarization words that tend to appear in passage summarization or taggings.

Table 1: Different kinds of term expansion.

Gating Controller

- Term expansion을 위한 4가지 scenario 중 Passage2query, Summarization에 집중
- passage-query, passage-summary 데이터만 가지고 2가지 CE loss 학습

$$L_{exp} = -\lambda_1 \sum_{j \in \{m|T_m=0\}} log(1 - G_j) - \lambda_2 \sum_{k \in \{m|T_m=1\}} log G_k$$

- G_j: target text t에 등장하지 않은 단어들의 확률 분포 -> 작아지도록 학습
- G_k: target text t에 등장하는 단어들의 확률 분포 -> 커지도록 학습

- p: passage, t: target text (query / summary)
- T: binary BoW vectors of t
- G: dense gating probability distribution for p

$$L = L_{rank} + L_{exp}$$

Experimental Setup, Training Details

- Dataset
  - MSMARCO[train] for train, MSMARCO[dev] for eval
- Training Details
  - Importance Predictor, Gating Controller 모두 BERT로 initialize, weight 공유 X
  - 먼저 Gating Controller fine-tuning (MSMARCO train, 50k steps)
  - 이후 Gating Controller 고정해두고 Triplet Loss로 SparTerm fine-tuning
- Indexing
  - Document를 SparTerm 모델에 통과시킨 sparse representation으로 inverted index
  - 검색 시, 들어오는 query를 SparTerm에 통과시켜 확장 후, 검색
    (즉, vocab 크기의 query sparse representation과 document sparse representation의 곱을 구함)

SparTerm

Experimental Setup, Training Details

- Baselines
  - BM25
  - DeepCT
  - Doc2query
  - Doc2query-T5: T5 기반으로 Doc2query 방법 그대로 한 것. SOTA였음
  - SparTerm (literal-only): passage에 제공된 단어만으로 gating controller 만든 것
  - SparTerm (expansion-only): passage에 제공되지 않은 단어만으로 gating controller를 만든 것
  - SparTerm (expansion-enhanced): passage에 제공된 단어와 제공되지 않은 단어를 모두 활용하여 gating controller 만든 것

SparTerm

Results

Model	MRR@10	R@10	R@20	R@50	R@100	R@200	R@500	R@1000
BM25	18.6	-	49	60	69	75	82	85.71
Doc2query	21.5	-	-	-	-	-	-	89.1
Doc2query-T5	27.7	-	-	75.6	81.89	86.88	91.64	94.7
DeepCT	24.3	49	58	69	76	82	86	91
SparTerm(literal-only)	27.46	51.05	60.21	71.55	78.28	83.27	88.33	91.16
SparTerm(expansion-only)	19.8	40.93	-	63.42	70.96	77.62	84.81	89.08
SparTerm(expansion-enhanced)	**27.94**	51.95	61.58	72.48	78.95	84.05	89.5	92.45

Table 2: Performances of different models on Dev Set of MSMARCO Passage Retrieval dataset.

- SparTerm이 MRR에서 가장 좋았고, RecallI에서 Doc2query-T5를 제외한 베이스라인 중 가장 높은 성능
- Doc2query-T5이 Doc2query(Transformer)보다 훨씬 좋은 것으로 보아, SparTerm도 좋은 모델로 바꾸면 더 잘할 것
- Expansion 없는, literal-only 모델이 DeepCT 이김

SparTerm

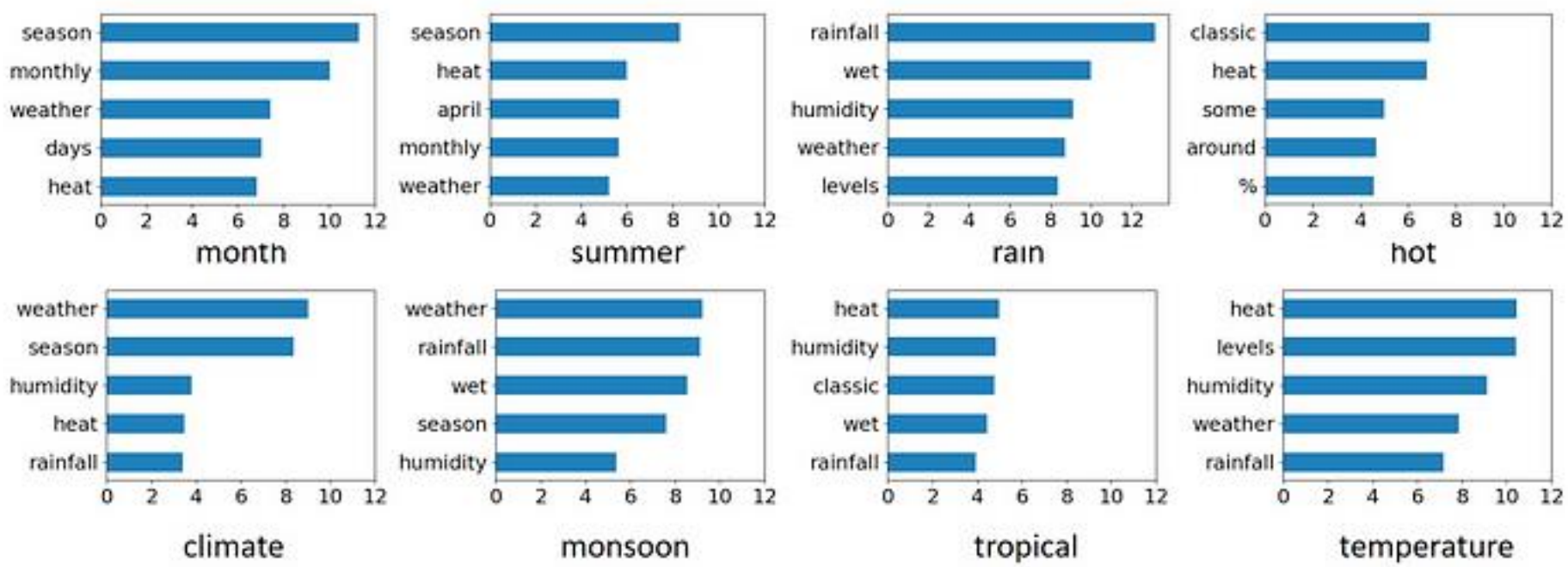
Qualitative Analysis

Query	Type	DeepCT	SPART
Can hives be a sign of pregnancy?	Literal term Weights	hives are caused by allergic reactions . the dryness and stretching of your skin along with other changes can make you more susceptible to experiencing hives during pregnancy . hives can be caused by an allergic reaction to almost anything . some common causes of hives during pregnancy are noted below : medicine .	hives are caused by allergic reactions . the dryness and stretching of your skin along with other changes can make you more susceptible to experiencing hives during pregnancy . hives can be caused by an allergic reaction to almost anything . some common causes of hives during pregnancy are noted below : medicine
	Term expansion		symptoms:1.0, women:0.99, rash:0.98, feel:0.99, causing:0.97, body:0.96, affect:0.96, baby:0.94, pregnant:0.93, **sign:0.91**, …

- DeepCT, SparTerm 모두 query가 입력되면 passage에 있는 단어들 중 가장 중요한 단어에 weight를 높게 매김
- 그러나 SparTerm이 DeepCT보다 조금 더 많은 단어들에 weight를 주기 때문에,
  "hives"가 들어오면 SparTerm은 "allergic reactions"같은걸 잡아낼 수 있음
- 또 query의 "sign"으로 "symptoms"라는 단어를 예측하고, "pregnancy"로 "women"이라는 단어를 예측

SparTerm

Qualitative Analysis



SPLADE: Sparse Lexical and Expansion Model for First Stage Ranking

Thibault Formal
Naver Labs Europe
Meylan, France
Sorbonne Université, LIP6
Paris, France
thibault.formal@naverlabs.com

Benjamin Piwowarski
Sorbonne Université, CNRS, LIP6
Paris, France
benjamin.piwowarski@lip6.fr

Stéphane Clinchant
Naver Labs Europe
Meylan, France
stephane.clinchant@naverlabs.com

SPLADE

Problem?

SparTerm(literal-only)	27.46	51.05	60.21	71.55	78.28	83.27	88.33	91.16
SparTerm(expansion-only)	19.8	40.93	-	63.42	70.96	77.62	84.81	89.08
SparTerm(expansion-enhanced)	**27.94**	51.95	61.58	72.48	78.95	84.05	89.5	92.45

- SparTerm 모델이 너무 복잡하고, end-to-end로 학습할 수 없다.
- SparTerm에 Gate Controller 추가해서 expansion-enhanced 버전 사용하는게 큰 성능 향상 X
  -> 굳이 필요한가 ?

SPLADE

Method

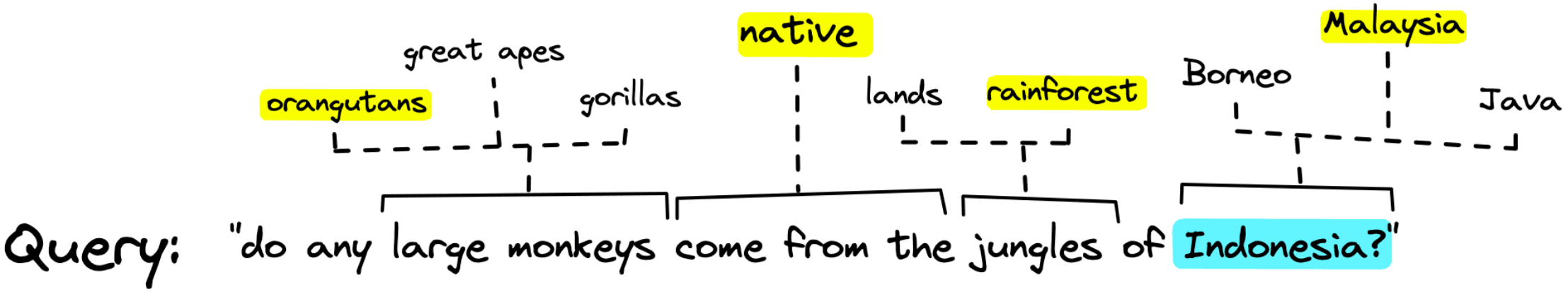
$$I = \sum_{i=0}^{L} Relu(I_i) \quad \longrightarrow \quad w_j = \sum_{i \in t} \log\left(1 + \mathrm{ReLU}(w_{ij})\right)$$

- Gating controller 없애고, Importance Predictor에서 log 하나만 추가 (SIGIR 2021 short accept)
- 직관적으로는, log을 붙임으로써 weight가 너무 큰 단어들을 조금 smoothing 할 수 있는 효과를 예상하게 됨
- Loss 식도

$$\mathcal{L}_{rank-IBN} = -\log \frac{e^{s(q_i, d_i^+)}}{e^{s(q_i, d_i^+)} + e^{s(q_i, d_i^-)} + \sum_j e^{s(q_i, d_{i,j}^-)}}$$

SPLADE

검색 상황 예시



with query expansion

without query expansion

Doc: "Orangutans are native to the rainforests of Indonesia and Malaysia"

SPLADE

Training details & Experimental Setup

- Dataset: MSMARCO[train] for train, MSMARCO[dev], TREC-2019[eval] for eval
- Training Details
  - Backbone: BERT-base
  - Batch_size = 124, 150k step동안 학습

SPLADE

Results

model	MS MARCO dev MRR@10	MS MARCO dev R@1000	TREC DL 2019 NDCG@10	TREC DL 2019 R@1000	FLOPS
Dense retrieval					
Siamese (ours)	0.312	0.941	0.637	0.711	-
ANCE [25]	0.330	0.959	0.648	-	-
TCT-ColBERT [15]	0.335	0.964	0.670	0.720	-
Sparse retrieval					
BM25	0.184	0.853	0.506	0.745	0.13
DeepCT [4]	0.243	0.913	0.551	0.756	-
doc2query-T5 [18]	0.277	0.947	0.642	0.827	0.81
ST lexical-only [1]	0.275	0.912	-	-	-
ST expansion [1]	0.279	0.925	-	-	-
Our methods					
ST lexical-only	0.290	0.923	0.595	0.774	1.84
ST exp-ℓ_1	0.314	0.959	0.668	0.800	4.62
ST exp-ℓ_{FLOPS}	0.312	0.954	0.671	0.813	2.83
SPLADE-ℓ_1	0.322	0.954	0.667	0.792	0.88
SPLADE-ℓ_{FLOPS}	0.322	0.955	0.665	0.813	0.73

- TREC DL Recall@1000을 제외하면 SPLADE가 모든 데이터, 모든 metric에서 퍼포먼스가 좋다.
- (당시) SOTA dense retriever과도 비교해도 견줄만하다.
- ST lexical-only를 InfoNCE 방법으로 학습한게 일반 ST expansion보다 높다.
  (InfoNCE vs. Triplet loss의 효과 비교)

SPLADE

Qualitative Analysis

original document (doc ID: 7131647)
if (1.2) bow (2.56) legs (1.18) ~~is~~ caused (1.29) by (0.47) ~~the~~ bone (1.2) alignment (1.88) issue (0.87) ~~than you may be~~ able (0.29) ~~to~~ correct (1.37) through (0.43) *bow legs* correction (1.05) ~~exercises~~. ~~read more here..~~ *if bow legs is caused by the bone alignment issue than you may be able to correct through bow legs correction exercises.*
expansion terms
(leg, 1.62) (arrow, 0.7) (exercise, 0.64) (bones, 0.63) (problem, 0.41) (treatment, 0.35) (happen, 0.29) (create, 0.22) (can, 0.14) (worse, 0.14) (effect, 0.08) (teeth, 0.06) (remove, 0.03)

- Model 통과 시 중요한 term에 대해서는 re-weighting, 불필요한 term (weight <=0)은 삭제
- Legs -> leg / bow legs -> treatment 으로 보아 expansion이 실제로도 효과 있음을 알 수 있음

SPLADE-v2: Pooling 방식의 변화

$$I = \sum_{i=0}^{L} Relu(I_i) \longrightarrow w_j = \sum_{i \in t} \log\left(1 + \mathrm{ReLU}(w_{ij})\right) \longrightarrow w_j = \max_{i \in t} \log\left(1 + \mathrm{ReLU}(w_{ij})\right)$$

SparTerm　　SPLADE　　SPLADE-v2

- SPLADE-v1에서는 sequence의 각 토큰 index에서 구한 vocab logit을 합침
- SPLADE-v2에서는 sequence의 각 토큰 index에서 구한 vocab logit 중 최대값 사용

SPLADE-v2: 결과

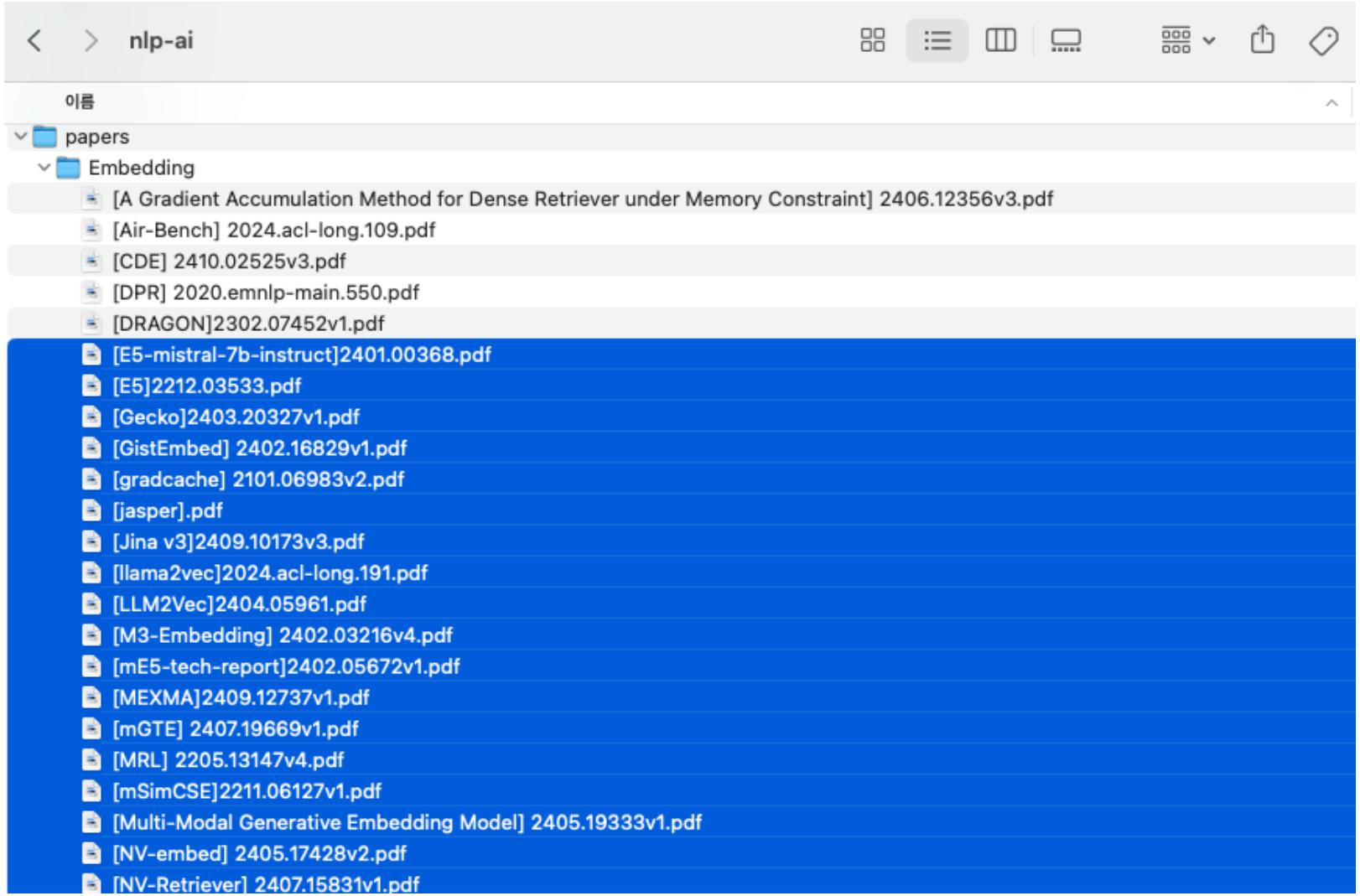
model	MS MARCO dev		TREC DL 2019	
	MRR@10	R@1000	NDCG@10	R@1000
Dense retrieval				
Siamese (ours)	0.312	0.941	0.637	0.711
ANCE [29]	0.330	0.959	0.648	-
TCT-ColBERT [16]	0.359	0.970	0.719	0.760
TAS-B [11]	0.347	0.978	0.717	0.843
RocketQA [24]	0.370	0.979	-	-
Sparse retrieval				
BM25	0.184	0.853	0.506	0.745
DeepCT [4]	0.243	0.913	0.551	0.756
doc2query-T5 [20]	0.277	0.947	0.642	0.827
SparTerm [1]	0.279	0.925	-	-
COIL-tok [9]	0.341	0.949	0.660	-
DeepImpact [18]	0.326	0.948	0.695	-
SPLADE [8]	0.322	0.955	0.665	0.813
Our methods				
SPLADE-max	0.340	0.965	0.684	0.851

Enhancing Lexicon-Based Text Embeddings with Large Language Models

Yibin Lei[1], Tao Shen[2], Yu Cao[3], Andrew Yates[1]
[1]University of Amsterdam [2]University of Technology Sydney [3]Tencent IEG
{y.lei, a.c.yates}@uva.nl, tao.shen@uts.edu.au,
rainyucao@tencent.com

LENS

SPLADE 등장 후..



LENS

LLM을 Sparse retrieval에 사용할 수 있을까 ?

문제는 Tokenizer

1. 같은 어휘소 (lexeme)으로부터 발생하는 중복성이 token matching에 악영향을 줄 수 있다는 문제 (e.g. “What”, “what”, “ what”이 모두 다른 token으로 처리되는 문제)
2. 한 단어들이 쪼개져서 추가적인 매칭이 이루어져야 한다는 문제 (e.g. “education” -> “edu”, “cation”)
3. 큰 문서 단위에서 학습된 tokenizer로 인해 rare한 term이 포함되어 vocab의 사이즈를 늘리는 문제

그럼 clustering해서 tokenizer를 줄여보자 !

LENS

LENS Framework

1. Tokenizer에 Kmeans clustering 진행 (k는 embedding 차원 수)
2. LM head의 token embeddings는 cluster의 centroid로 대체
3. Input text 가 tokenizer를 통해 변환된 후 생성되는 초기 embedding은 그대
   모델 내부에서 LM head의 output embedding만 clustering을 통해 간소화

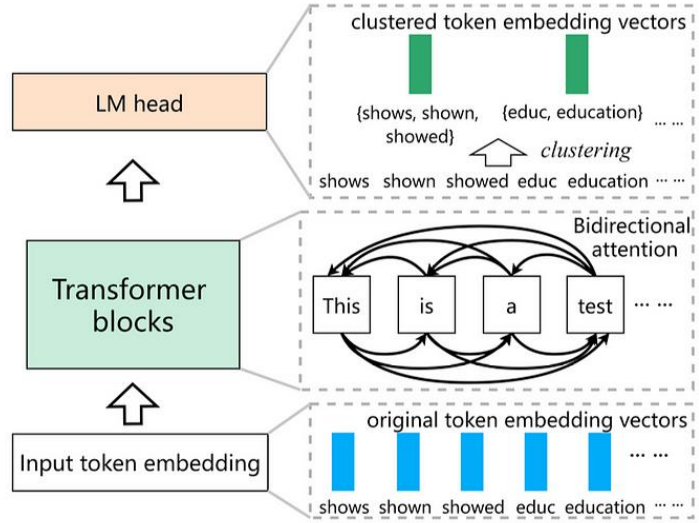


Figure 2: The model framework of LENS.

결과 – vocab size의 embedding 대신, clustered size의 lexicon-based embedding

LENS

Representation Generation

LLM으로 Representation을 어떻게 만들까?

1. Test set에서, query를 다음과 같이 q_ins 형태로 만듦 $q_{\text{ins}} = \langle \text{Instruct} \rangle \{\text{task\_definition}\} \langle \text{query} \rangle \{q\}.$
   (e.g.) "Given a question, retrieve passages that answer the question. Question: Who is Heuiseok Lim ?"
2. SPLADE-v2 처럼 max pooling 방식으로 pooling
3. 주의사항
   - "Query"에 대한 logit만 뽑아야함.
   - Logit을 뽑는 것인데, LLM은 Next Token Prediction 방식이므로, {보고자 하는 token의 index – 1}번째를 본다.
   - passage는 따로 instruction을 주지 않으니, 바로 BOS 토큰부터 생성되는 logit을 본다.

LENS

Training

InfoNCE Loss로 학습

$$\mathcal{L} = -\log \frac{\exp(\text{sim}(q_{\text{ins}}, p)/\tau)}{\exp(\frac{\text{sim}(q_{\text{ins}}, p)}{\tau}) + \sum_{j=1}^{N} \exp(\frac{\text{sim}(q_{\text{ins}}, p_j^-)}{\tau})}$$

q_ins와 p의 embedding (cluster 크기)를 뽑아 위 Loss로 학습

LENS

Experiment & Training Config

- Backbone model: Mistral-7B-v0.1
- Data: BGE-en-ICL에서 썼던 공개된 데이터- retrieval, reranking, clustering, classification, STS (1 positive, 7 hard negative)
- KMeans의 cluster 수: 4000, 8000 (LENS-4000, LENS-8000) -> 곧 차원 수
- Training config
  - LoRA 사용
  - 1 positive, 7 hard negative 사용
  - in-batch negative 사용 (retrieval task에 대한 batch size 512, 나머지 task에 대한 batch size 256)
  - BGE-reranker로부터 ranking score distillation. (KL-divergence. 이건 BGE-en-ICL 학습 방법 그대로 따라하려고 했다함)
  - max_seq_len=512
- Evaluation: MTEB, AIR-Bench (Jina에서 미공개 데이터로 1달에 한번씩 갱신하는 평가 벤치마크)

LENS

Experiment & Training Config

- Baselines
  - E5-mistral-7b-instruct
  - NV-Embed-v1/v2
  - gte-Qwen2–7B-instruct
  - Stella
  - SFR-Embedding-2_R
  - GritLM-7B
  - BGE-en-ICL
  - SPLADE 같은 모델들은 retrieval task 용도로만 만들어졌고, 너무 못해서 안넣음
- zero-shot scenario만 가정

LENS

Results: MTEB

Task # of datasets →	#Dims	Retr. 15	Rerank. 4	Clust. 11	PairClass. 3	Class. 12	STS 10	Summ. 1	Avg. 56
Non-Fully Public Training Data									
E5-mistral-7b-instruct	4096	56.90	60.21	50.26	88.34	78.47	84.66	31.40	66.63
Linq-Embed-Mistral	4096	60.19	60.29	51.42	88.35	80.20	84.97	30.98	68.17
voyage-large-2-instruct	1024	58.28	60.09	53.35	89.24	81.49	84.31	30.84	68.23
stella_en_400M_v5	8192	58.97	60.16	56.70	87.74	86.67	84.22	31.66	70.11
gte-Qwen2-7B-instruct	3584	60.25	61.42	56.92	85.79	86.58	83.04	31.35	70.24
SFR-Embedding-2_R	4096	60.18	60.14	56.17	88.07	89.05	81.26	30.71	70.31
stella_en_1.5B_v5	8192	61.01	61.21	57.69	88.07	87.63	84.51	31.49	71.19
NV-Embed-v2	4096	62.65	60.65	58.46	88.67	90.37	84.31	30.70	72.31
Fully Public Training Data									
LLM2Vec-Mistral-supervised	4096	55.99	58.42	45.54	**87.99**	76.63	84.09	29.96	64.80
GritLM-7B	4096	57.41	60.49	50.61	87.16	79.46	83.35	30.37	66.76
NV-Embed-v1	4096	59.36	60.59	52.80	86.91	87.35	82.84	**31.20**	69.32
bge-multilingual-gemma2	3584	59.24	59.72	54.65	85.84	88.08	83.88	**31.20**	69.88
BGE-en-ICL (zero-shot)	4096	61.67	59.66	57.51	86.93	**88.62**	83.74	30.75	71.24
LENS-4000 (***Ours***)	4000	60.76	60.86	57.92	87.93	88.13	84.35	31.56	71.21
LENS-8000 (***Ours***)	8000	**61.86**	**60.91**	**58.02**	87.98	88.43	**84.67**	29.54	**71.62**

LENS

Results: Air-Bench

Domain # of datasets →	#Dims	wiki 1	web 1	news 1	healthcare 1	law 1	finance 1	arxiv 1	msmarco 1	Avg. 8
E5-mistral-7b-instruct	4096	61.67	44.41	48.18	56.32	19.32	54.79	44.78	59.03	48.56
Linq-Embed-Mistral	4096	61.04	48.41	49.44	**60.18**	20.34	50.04	47.56	60.50	49.69
NV-Embed-v1	4096	62.84	50.42	51.46	58.53	20.65	49.89	46.10	60.27	50.02
gte-Qwen2-7B-instruct	3584	63.46	51.20	54.07	54.20	22.31	**58.20**	40.27	58.39	50.26
stella_en_1.5B_v5	8192	61.99	50.88	53.87	58.81	23.22	57.26	44.81	61.38	51.53
SFR-Embedding-Mistral	4096	63.46	51.27	52.21	58.76	23.27	56.94	47.75	58.99	51.58
NV-Embed-v2	4096	65.19	52.58	53.13	59.56	25.00	53.04	**48.94**	60.80	52.28
BGE-en-ICL (zero-shot)	4096	64.61	54.40	55.11	57.25	25.10	54.81	48.46	**63.71**	**52.93**
LENS-4000 (***Ours***)	4000	62.60	52.06	52.49	57.23	24.08	48.87	43.78	61.17	50.28
LENS-8000 (***Ours***)	8000	**65.50**	**54.52**	**55.16**	58.20	**25.62**	54.57	45.45	63.00	52.75

LENS

Qualitative Analysis

Clusters
quickly, rapid, rapidly, swift
cannot, impossible, Unable, Cannot, Unable
shows, shown, showed, showing
review, Review, reviews, reviewed, Reviews
educ, education, Educ, Education, educational, Edu

LENS

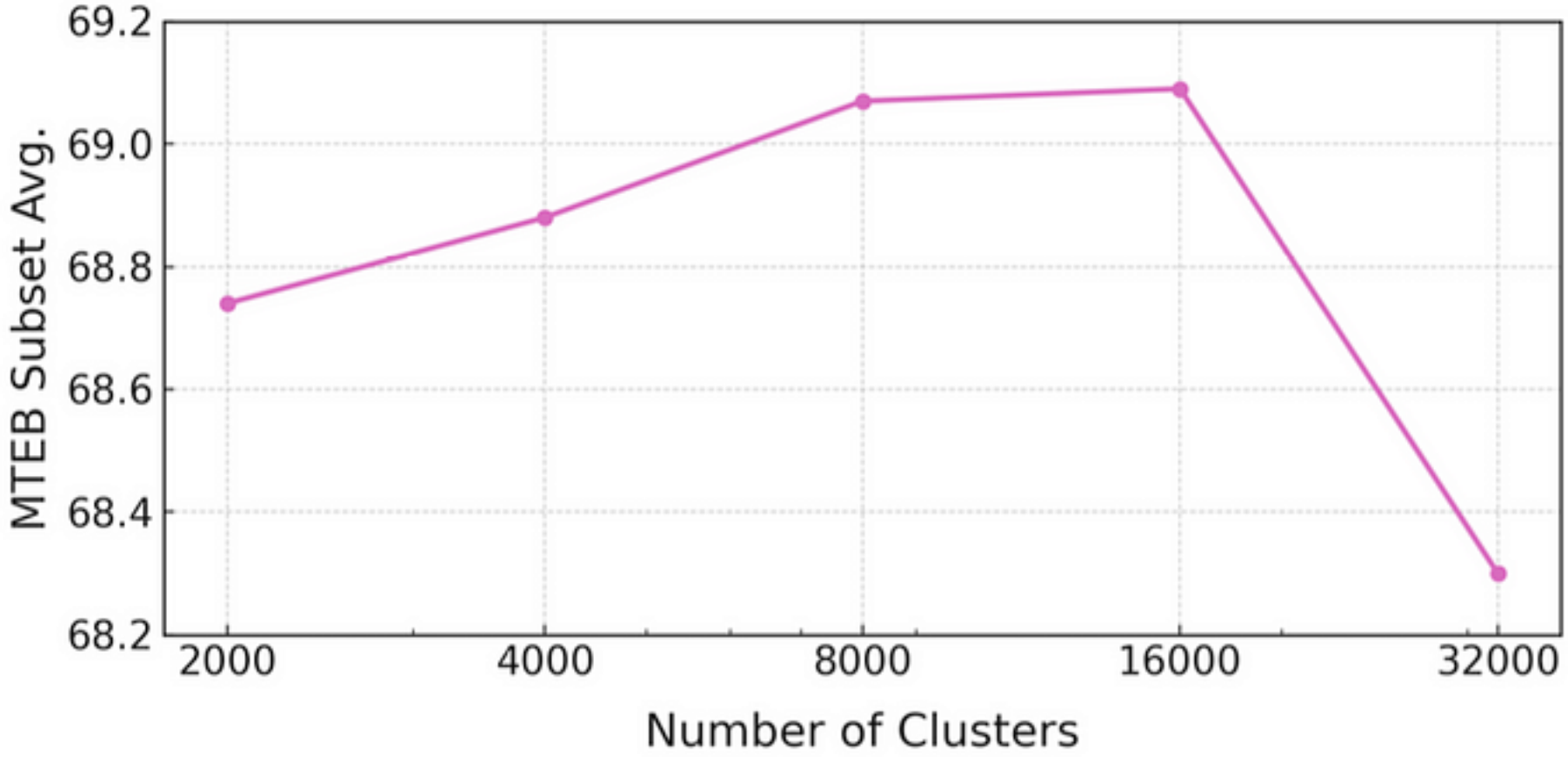
Qualitative Analysis

Text	Top-weighted clusters
most dependable affordable cars	(cars, Cars), (cheap, affordable), (reliable, reli), (depend, depends), (aff, afford)
fastest growing bonsai trees	(faster, fastest), (grow, growing), (fast, Fast), (tree, trees), (quickly, rapid)
causes of hypoxia in adults	(adult, adults), (oxygen, oxy), (cause, caused), (hyp, yp), (ox, Ox)
weather in lisbon april	(Portug, Portuguese), (bon, Bon), (weather, rather), (Spring, spring), (AP, #AP)
other hot flashes causes	(hot, Hot), (cause, causes), (flash, Flash), (flush, #flush), (heat, Heat)

Table 3: Qualitive examples of LENS-8000. For each example, the top-5 clusters with the largest weights in the embeddings are shown, with two tokens from each cluster included.

LENS

Analysis – Cluster에 따른 성능



LENS

Analysis – Pooling, Attention에 따른 성능

Task # of datasets →	Retr. 1	Rerank. 1	Clust. 1	PairClass. 1	Class. 1	STS 1	Summ. 1	Avg. 7
Unidirectional Attention								
Last-token pooling	73.84	65.19	60.46	96.69	58.66	89.26	30.05	67.73
Sum-pooling	72.46	59.57	50.55	89.90	54.64	80.55	29.70	62.48
Max-pooling	75.18	59.68	50.93	92.06	57.58	82.74	30.89	64.15
Bidirectional Attention								
Last-token pooling	76.89	64.21	61.57	96.62	58.33	88.72	30.72	68.15
Sum-pooling	75.65	63.64	61.77	96.97	60.05	89.58	30.98	68.38
Max-pooling	76.19	64.53	63.05	97.03	62.30	88.92	31.49	69.07

Table 5: Influence of attention mechanisms and pooling methods.

(개인적으로 생각하는) 검색의 미래

1. Agent 시대로 갈수록, 검색이 더 중요해지지 않을까
2. 검색을 위해 문맥적 의미로 검색하는 Dense retrieval도 있지만, sparse retrieval도 강력한 베이스라인임.
   앞으로 LLM의 logit에 기반한 sparse retrieval이 더 많이 연구될 것 같다. (특히 multilingual)
3. Retriever와 마찬가지로 Reranker 또한 작년 후기부터 많은 이목을 끌었다.
   (기업의 ROI; Return Of Investment 을 고려한다면 retriever로 top 100을 가져와 100개로 reranking 하는 것이 비용적인 측면에서 훨씬 더 효율이 좋기 때문이다.)
4. Cross-lingual retrieval에 대한 니즈

Thank you

Q&A