Nota 기업 연계 프로젝트

NLP-06

팀명 : 어쩔돌멩이

팀원 : 김민호 김성은 김지현 서가은 홍영훈

boostcamp ai tech

INDEX

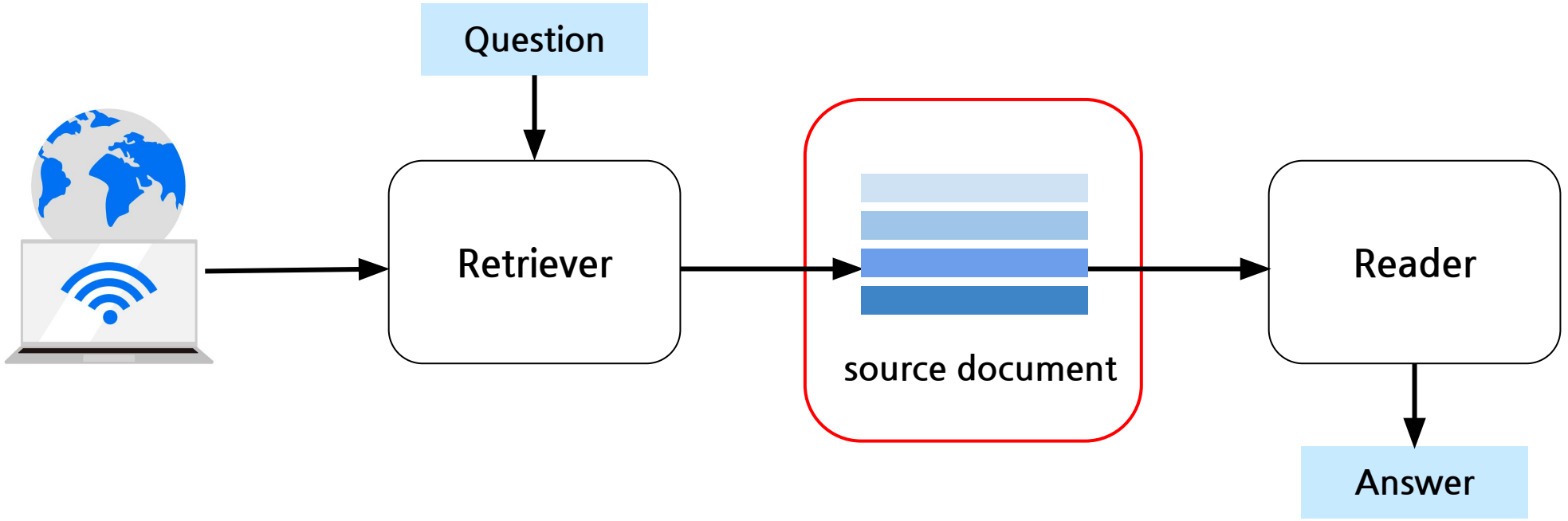
1. Intro

문제 정의 & 주제 제안

팀 및 팀원 소개

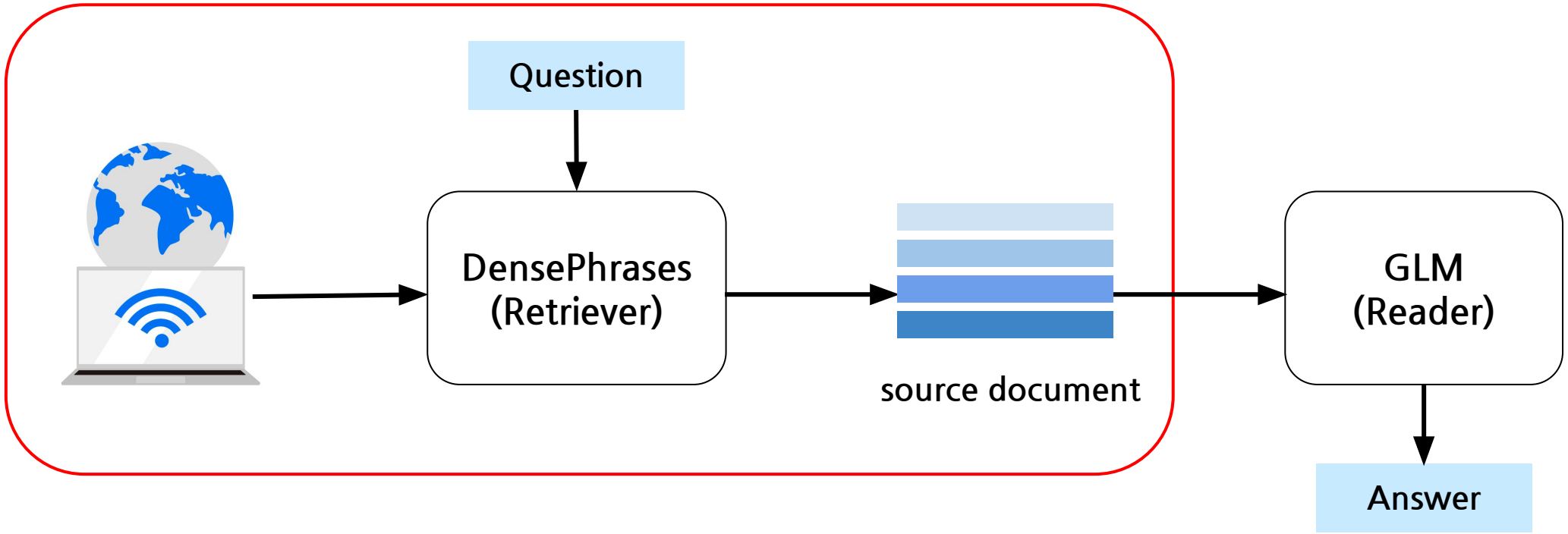
Timeline

1.1. 문제 정의 & 주제 제안

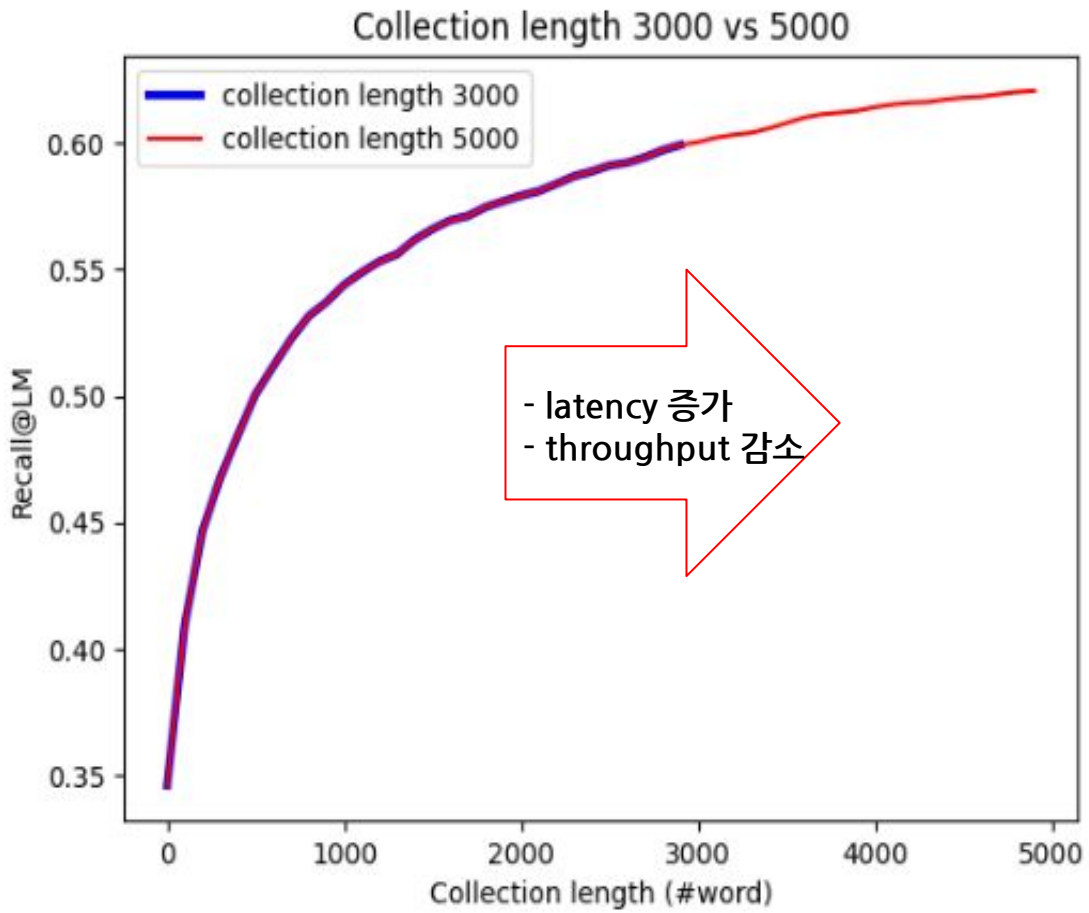


ODQA(Open-domain Question Answering)
질의가 주어지면 주어진 질의에 답할 수 있는 문장들을 Retriever가 지식 베이스로부터 찾아 근거 문서를 구성하고, 구성된 근거 문서(source document)를 기반으로 Reader가 답변하는 시스템

Project Overview



Recall@LM과 collection length(근거 문서의 길이)

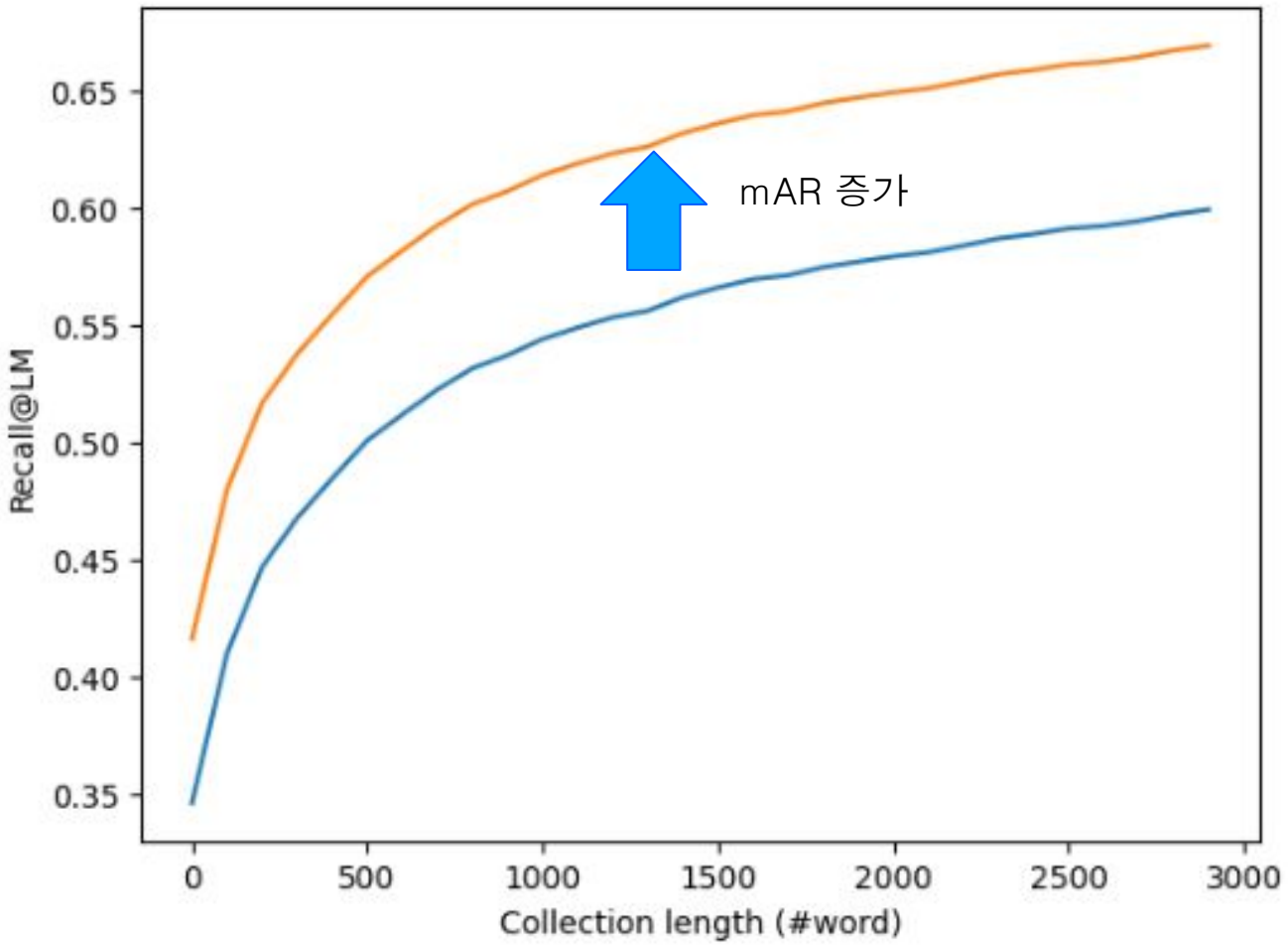


Reader에게 전달되는 **근거 문서(prompt)** 길이 증가

-> **정답 포함율(Recall@LM) 증가**

-> **BUT 정답 생성에 필요한 비용 역시 증가**

- **프로젝트 목표 : mAR 증가를 통한 검색 최적화**



- mAR : 근거 문서의 길이에 따른 평균 정답 포함율로, 정답 생성에 필요한 비용과 정답 포함율을 모두 고려한 지표

$$mAR = \frac{\sum_{i=1}^{L} Recall@LM(i)}{L}$$

$$Recall@LM(i) = \frac{\sum_{q \in Q} recall(q, i)}{n(Q)}$$

$$L = max\ collection\ length$$

- mAR을 높이기 위해서는 **같은 collection length**에서 **더 높은 Recall@LM**을 가져야 함

- 본 프로젝트에서는 mAR을 높이기 위해 두 가지 측면에서 접근
  - **retrieval 알고리즘 수정**
  - **Query Side Fine-tuning 수행**

1.2. 팀 및 팀원 소개

김민호

- Query loss의 단위 변경
- Loss의 구성 요소 추가

김성은

- Query loss의 단위 변경
- Dynamic retrieval

김지현

- Dataset 전처리
- Query loss의 단위 변경
- Knowledge Distillation

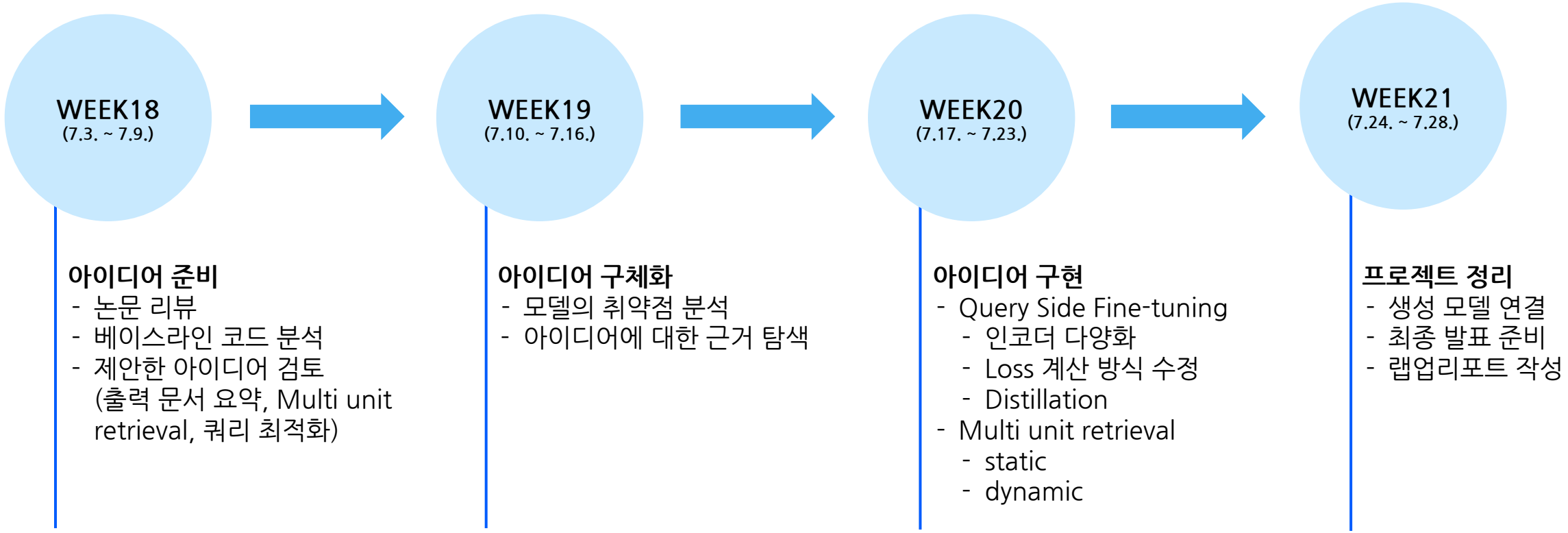
서가은

- Query loss의 단위 변경
- Dynamic retrieval

홍영훈

- Query loss의 단위 변경
- Static retrieval
- Optimization

1.3. Timeline



2. dataset, baseline model 소개

Dataset

Baseline model

2.1. Dataset

NQ (Natural Question) Dataset : QA (Question Answering) Task의 Benchmark Dataset 중 하나.

- id : 질문 id
- question : 질문 텍스트
- titles : 질문의 답이 속한 문서 제목
- sentence : 정답들이 속한 문장 리스트
- context : 정답들이 속한 문단
- answers : 정답들로 구성된 리스트

id	2126085010748850659
question	when does season 5 of bates motel come out
titles	Bates Motel (season 5)
sentence	[' The fifth and final season of "Bates Motel" premiered on **February 20, 2017**, and concluded on April 24, 2017.']
context	' The fifth and final season of "Bates Motel" premiered on February 20, 2017, and concluded on April 24, 2017. The season consisted of 10 episodes and aired on Mondays at 10 p.m. ET/PT on A&E. The series itself is described as a "contemporary prequel" to the 1960 film "Psycho", following the life of Norman Bates and his mother Norma prior to the events portrayed in the Hitchcock film. However, the final season of the series loosely adapts the plot of "Psycho". The series takes place in the fictional town of White Pine Bay, Oregon.'
answers	[February 20, 2017]

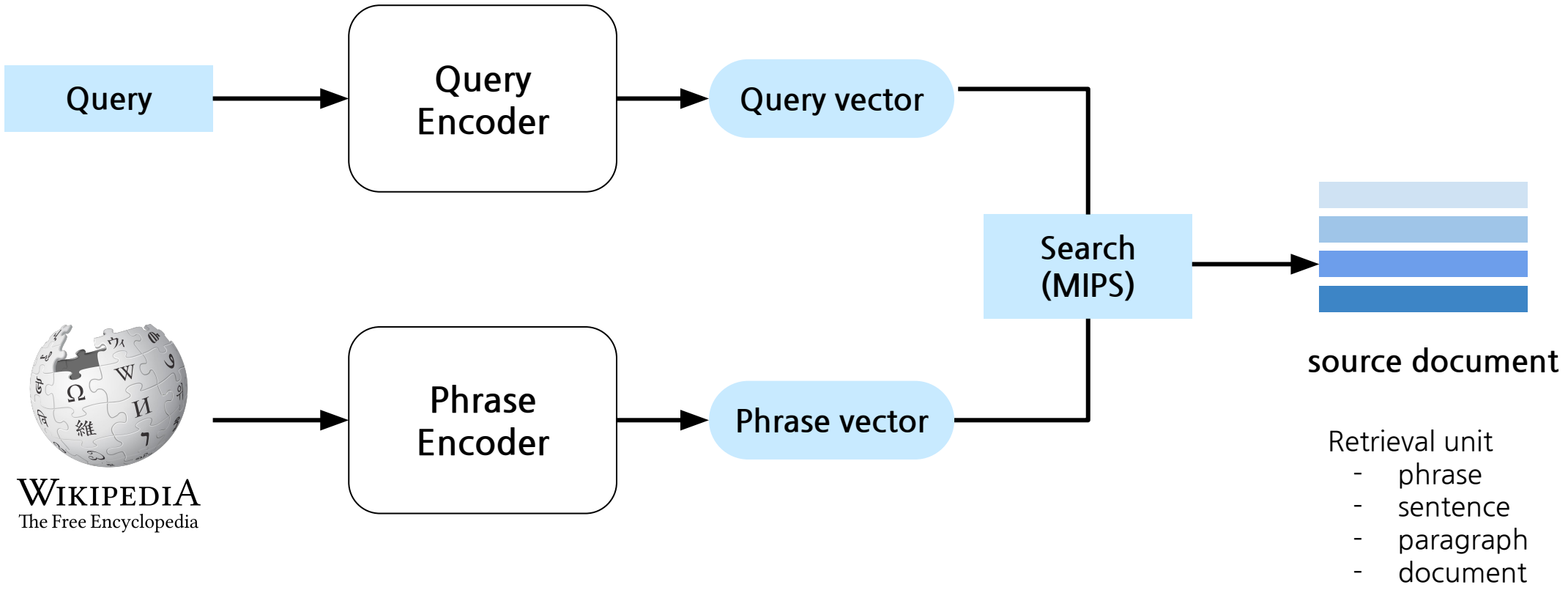
2.1. Dataset

NQ (Natural Question) Dataset : QA (Question Answering) Task의 Benchmark Dataset 중 하나.

- id : 질문 id
- question : 질문 텍스트
- titles : 질문의 답이 속한 문서 제목
- sentence : 정답들이 속한 문장 리스트
- context : 정답들이 속한 문단
- answers : 정답들로 구성된 리스트

id	2126085010748850659
question	when does season 5 of bates motel come out
titles	Bates Motel (season 5)
sentence	[' The fifth and final season of "Bates Motel" premiered on **February 20, 2017**, and concluded on April 24, 2017.']
context	' The fifth and final season of "Bates Motel" premiered on **February 20, 2017**, and concluded on April 24, 2017. The season consisted of 10 episodes and aired on Mondays at 10 p.m. ET/PT on A&E. The series itself is described as a "contemporary prequel" to the 1960 film "Psycho", following the life of Norman Bates and his mother Norma prior to the events portrayed in the Hitchcock film. However, the final season of the series loosely adapts the plot of "Psycho". The series takes place in the fictional town of White Pine Bay, Oregon.'
answers	[February 20, 2017]

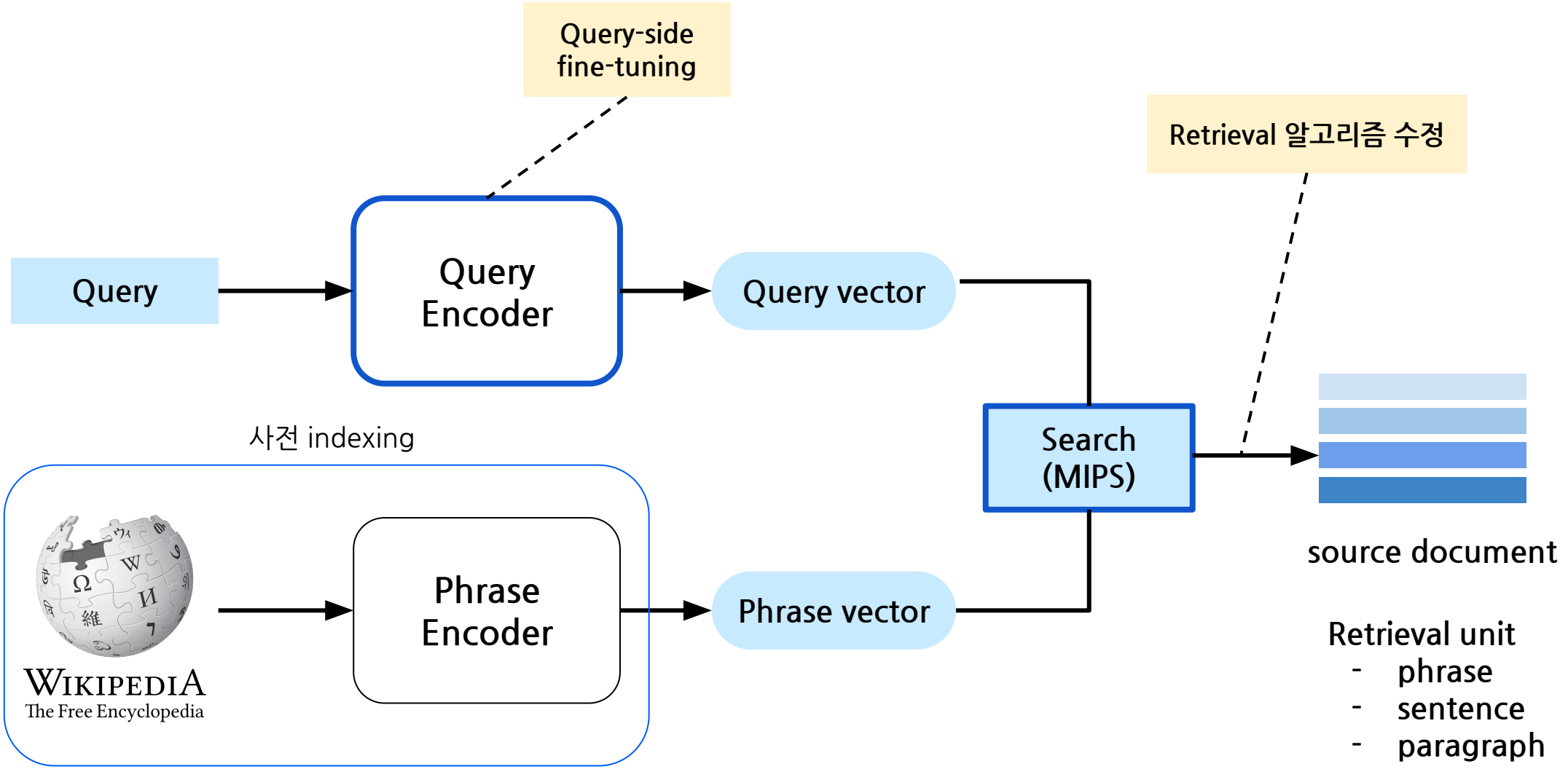
DensePhrases



3. Model & Research

QSFT(Query Side Fine-tuning)

Multi Unit Retrieval



3.1. QSFT (Query Side Fine-tuning)

- Query loss의 단위 변경
- Loss의 구성 요소 추가
- Knowledge Distillation

Query loss의 단위 변경

QSFT 단계의 loss 수정

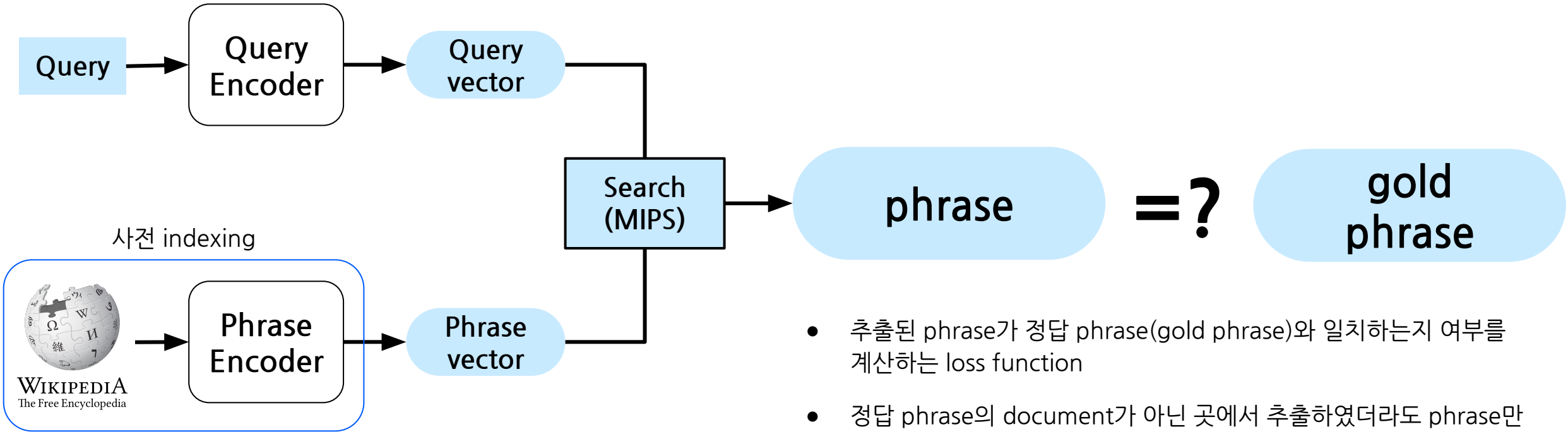
1. $$\mathcal{L}_{\text{query}} = -\log \frac{\sum_{s \in \tilde{\mathcal{S}}(q), \texttt{TEXT}(s)=a^*} \exp\left(f(s|\mathcal{D}, q)\right)}{\sum_{s \in \tilde{\mathcal{S}}(q)} \exp\left(f(s|\mathcal{D}, q)\right)},$$

2. $$\mathcal{L}_{\text{doc}} = -\log \frac{\sum_{s \in \tilde{\mathcal{S}}(q), d(s) \in \mathcal{D}^*} e^{f(s,q)}}{\sum_{s \in \tilde{\mathcal{S}}(q)} e^{f(s,q)}},$$

3. $$L_{sentence} = -log \frac{\sum_{s \in \tilde{S}(q), SENTENCE(s) \int sentence^*} e^{f(s,q)}}{\sum_{s \in \tilde{S}(q)} e^{f(s,q)}}$$

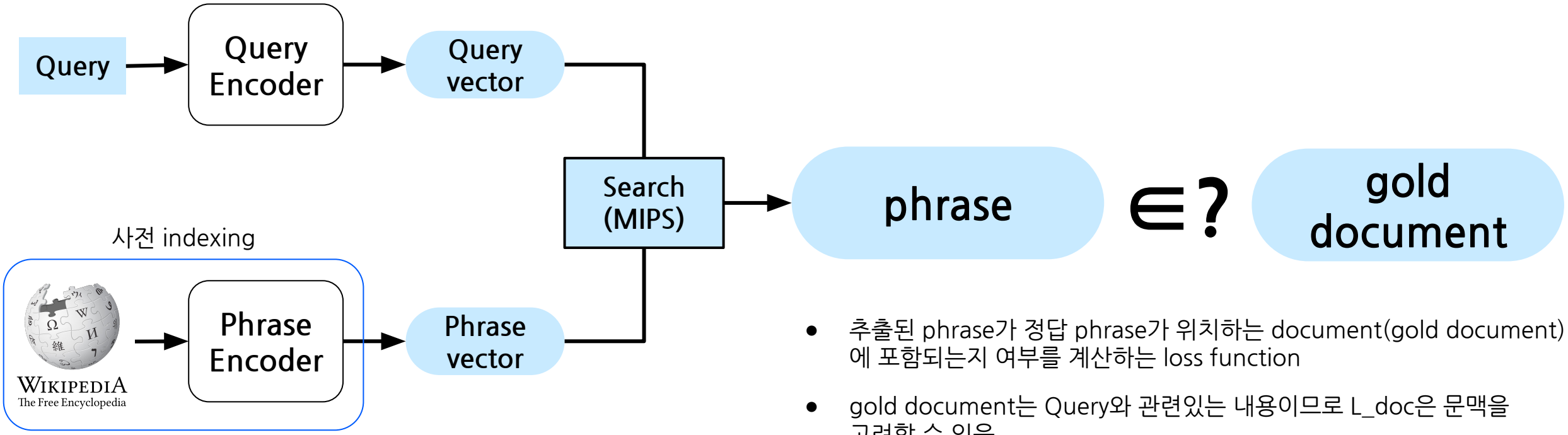
4. $$L_{context} = -log \frac{\sum_{s \in \tilde{S}(q), CONTEXT(s) \int context^*} e^{f(s,q)}}{\sum_{s \in \tilde{S}(q)} e^{f(s,q)}}$$

Query loss의 단위 변경: L_query



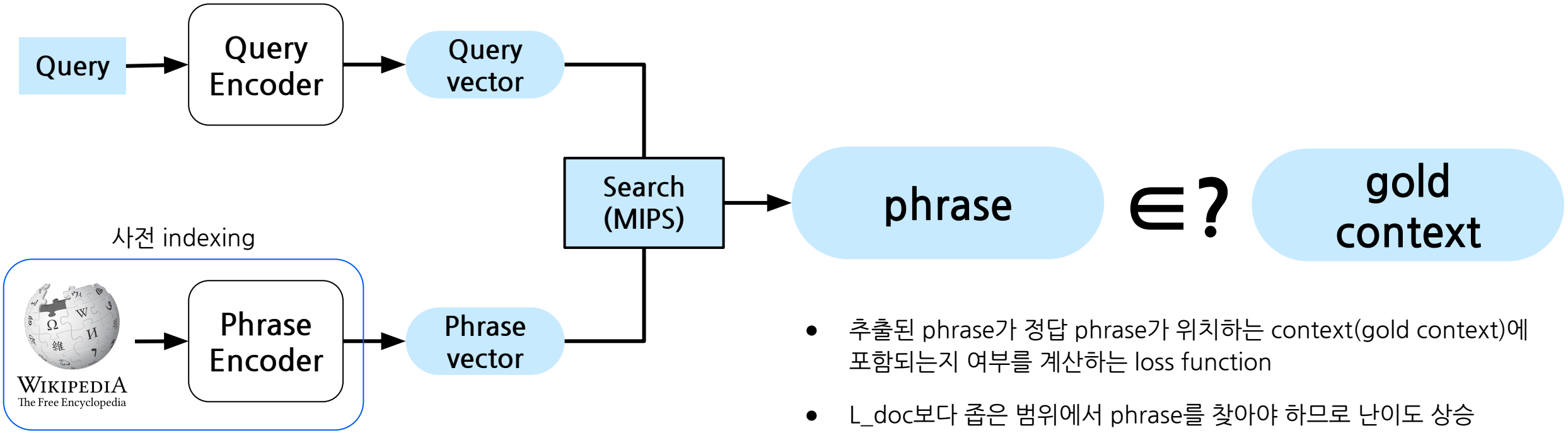
- 추출된 phrase가 정답 phrase(gold phrase)와 일치하는지 여부를 계산하는 loss function
- 정답 phrase의 document가 아닌 곳에서 추출하였더라도 phrase만 일치하다면 정답으로 간주

Query loss의 단위 변경: L_doc



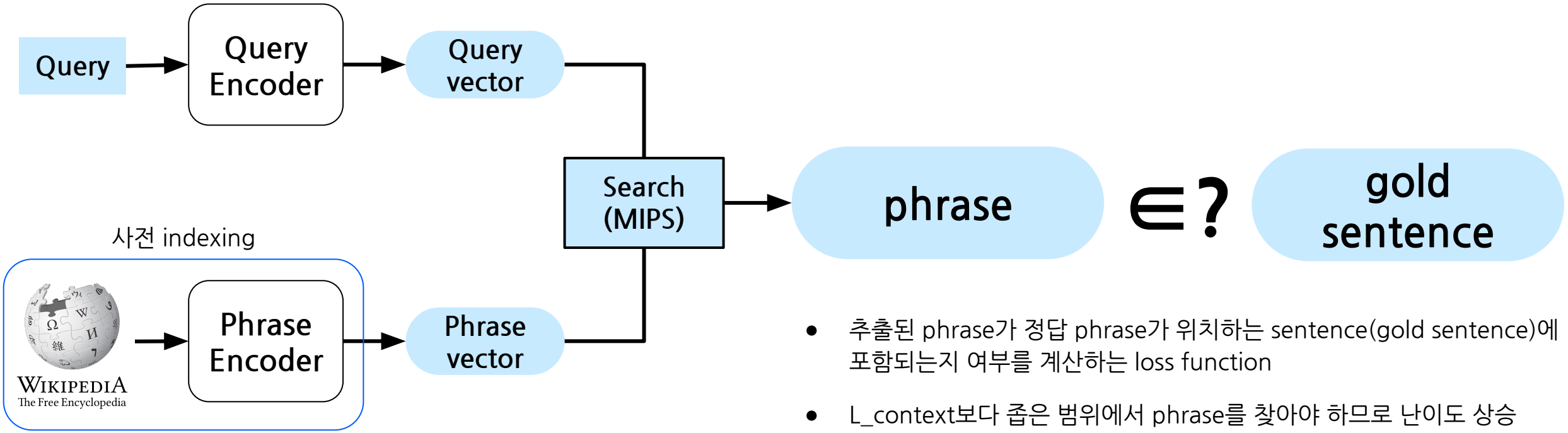
- 추출된 phrase가 정답 phrase가 위치하는 document(gold document)에 포함되는지 여부를 계산하는 loss function
- gold document는 Query와 관련있는 내용이므로 L_doc은 문맥을 고려할 수 있음
- 정확한 phrase를 맞추지 못하였더라도 정확한 document에서 추출하였는지를 감지

Query loss의 단위 변경: L_context



- 추출된 phrase가 정답 phrase가 위치하는 context(gold context)에 포함되는지 여부를 계산하는 loss function
- L_doc보다 좁은 범위에서 phrase를 찾아야 하므로 난이도 상승

Query loss의 단위 변경: L_sentence

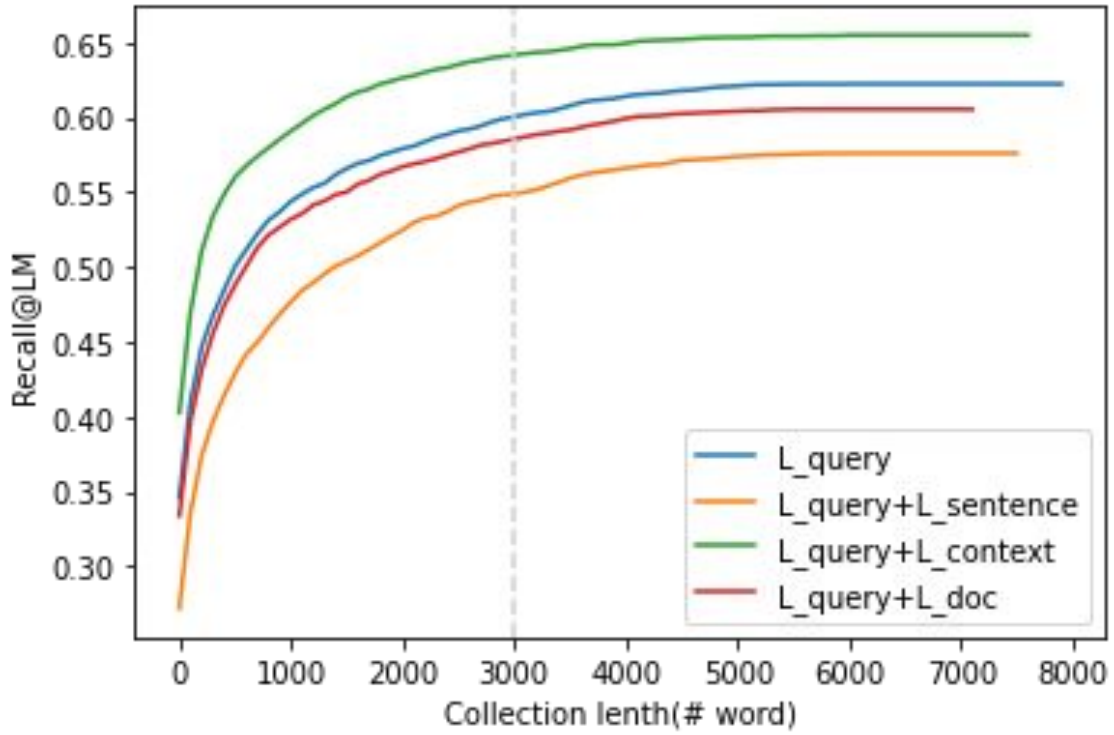


- 추출된 phrase가 정답 phrase가 위치하는 sentence(gold sentence)에 포함되는지 여부를 계산하는 loss function
- L_context보다 좁은 범위에서 phrase를 찾아야 하므로 난이도 상승

Query loss의 단위 변경: Result

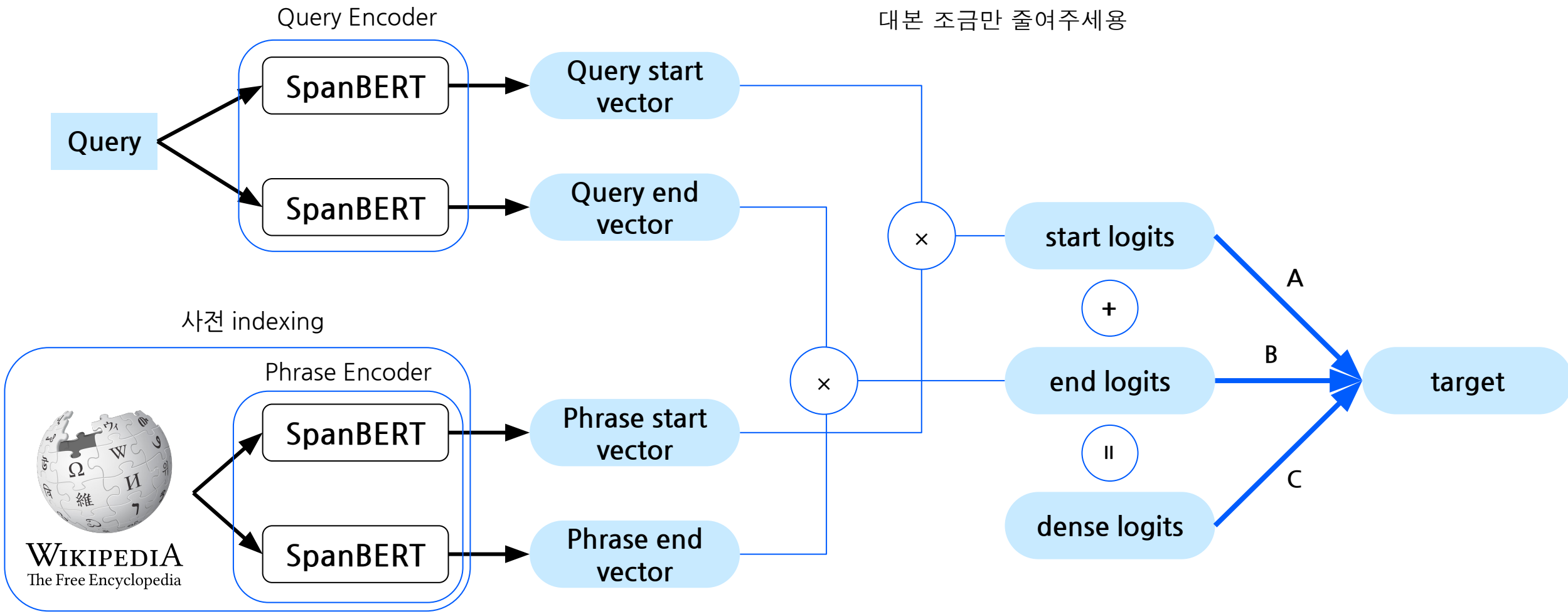
R_unit : sentence
Top_k : 200
Collection length = 3000

수정 필요

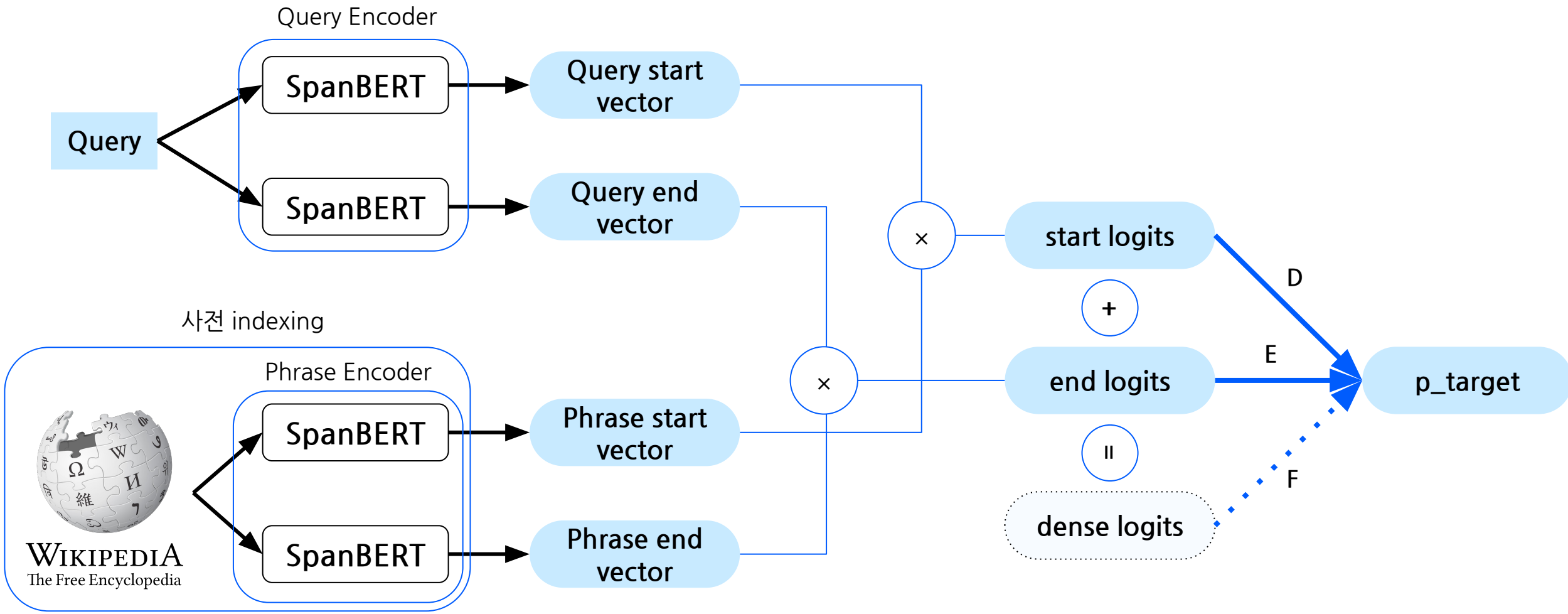


Loss	mAR
L_query	0.6316
L_query + L_sentence	
L_query + L_context	0.6322
L_query + L_doc	0.6351

Loss의 구성 요소 추가



Loss의 구성 요소 추가



Loss의 구성 요소 추가

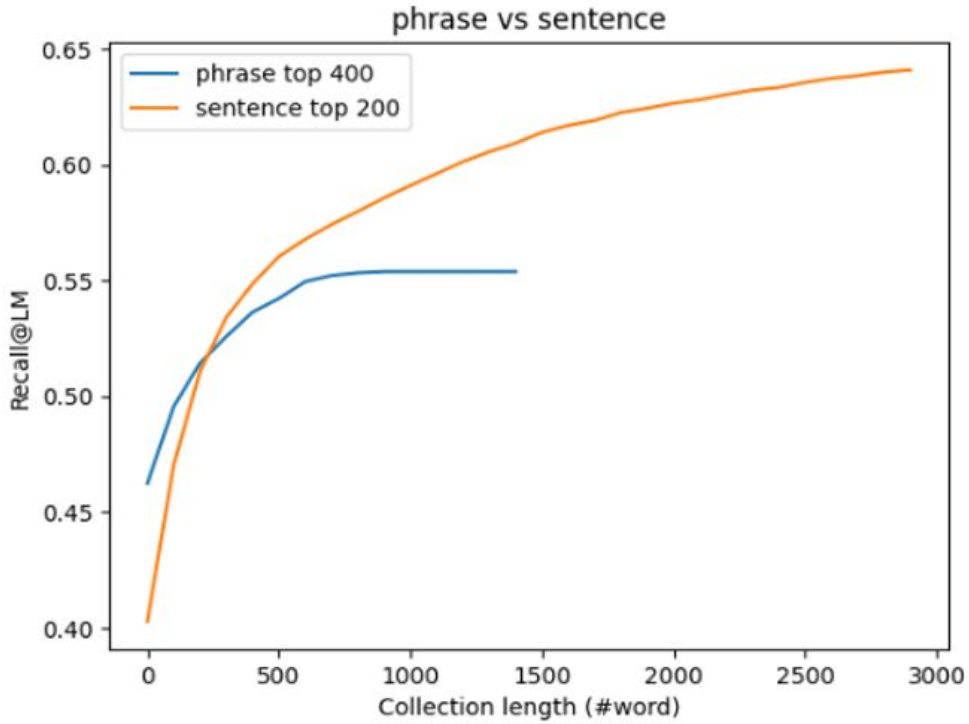
- 결과
  - phrase_correct: 출력된 sentence에 정답이 포함되어 있고, 그 정답이 해당 sentence를 출력하게 한 phrase일 경우
  - sent_correct: 출력된 sentence에 정답이 포함되어 있지만 그 정답이 해당 sentence를 출력하게 한 phrase는 아닐 경우

	phrase_correct	sent_correct	sum	mAR	Loss의 구성 요소
L_query	**2,505**	1,773	4,278	0.6316	A + B + C
L_query+L_doc	2,385	1,925	**4,310**	**0.6353**	A + B + C + D + E
L_query+L_doc custom	2,302	**1,993**	4,295	0.6332	A + B + C + D + E + F

	phrase_correct	sent_correct	sum	mAR	Loss의 구성 요소
L_query	**2,505**	1,773	4,278	0.6316	A + B + C
L_query+L_context	2,105	2,187	**4,292**	**0.6322**	A + B + C + D + E
L_query+L_context custom	1,941	**2,323**	4,264	0.6289	A + B + C + D + E + F

Knowledge Distillation (with cross-encoder model)

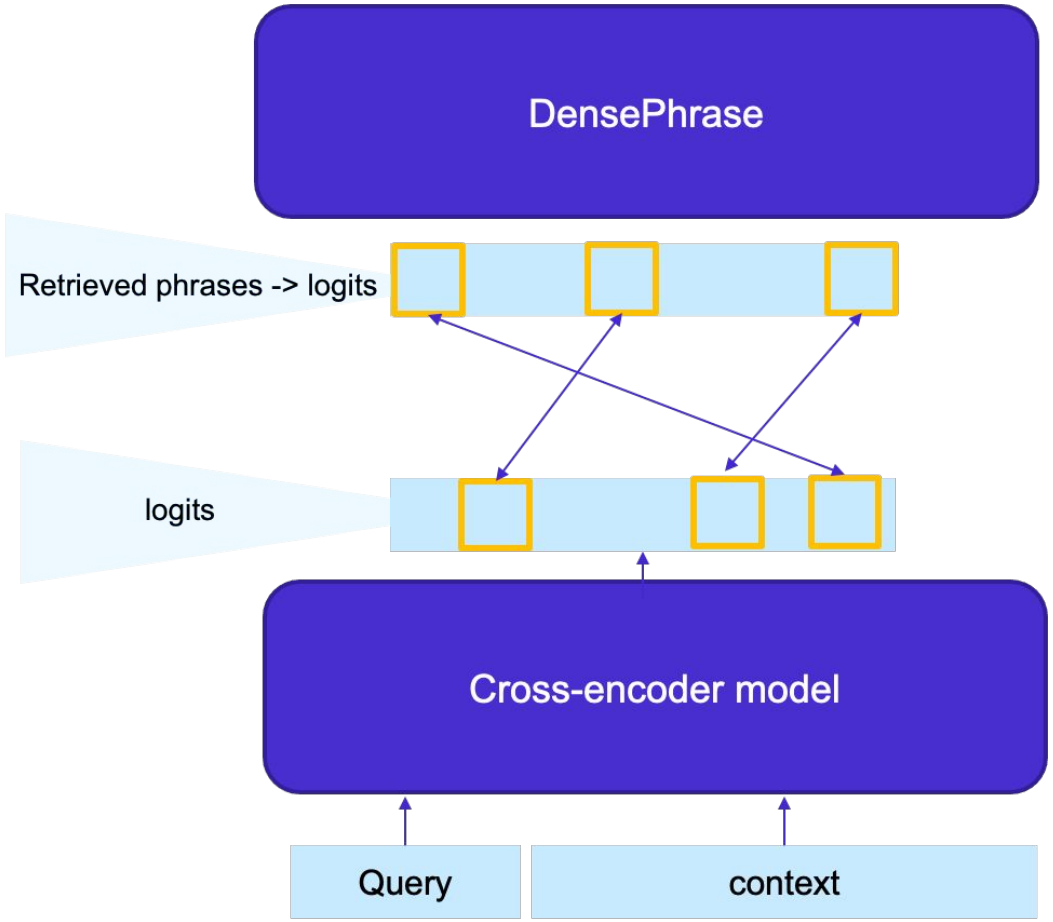
: Cross-encoder model 에서 생성한 logit 과 densephrase 에서 생성한 Logit 의 차이를 loss 로 사용



-
- context 내에서 gold phrase 의 logit 값을 높이는 것
-

Knowledge Distillation (with cross-encoder model)

1. 정답 context 와 query 를 cross-encoder model 에 input 으로 넣어 logit 을 생성
2. Densephrase 에서 retrieve 된 top-k개의 phrase 들중 정답 context 에 포함된 phrase 들의 위치를 추출
3. 2에서 추출 된 phrase token 들의 logit 값과 1에서 생성된 logit 값을 비교하여 그 차이를 distillation loss 로 사용

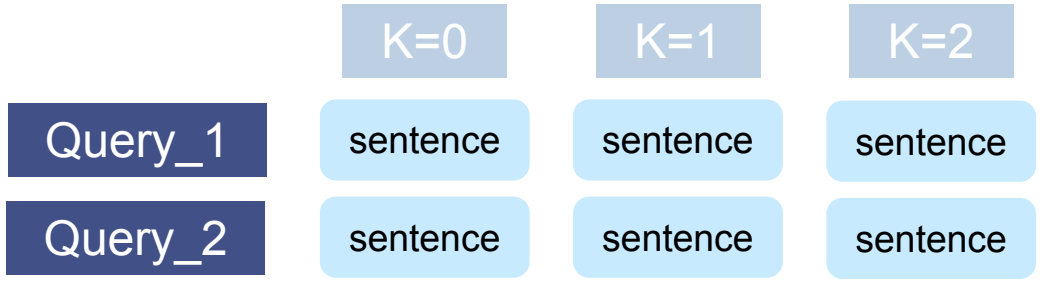


3.2 Multi Unit Retrieval

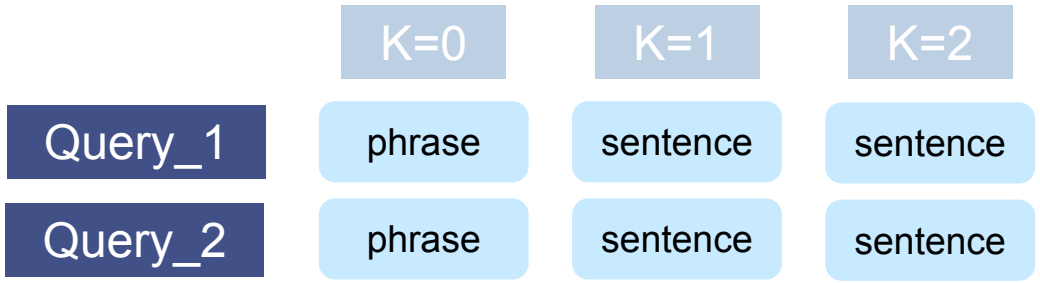
- **Static**
- **Dynamic** Multi encoder -> Dual query encoder 로 변경
- **Optimization**
- **Result** Multi unit retrieval -> 여러가지 실험 진행 결과

Multi Unit Retrieval

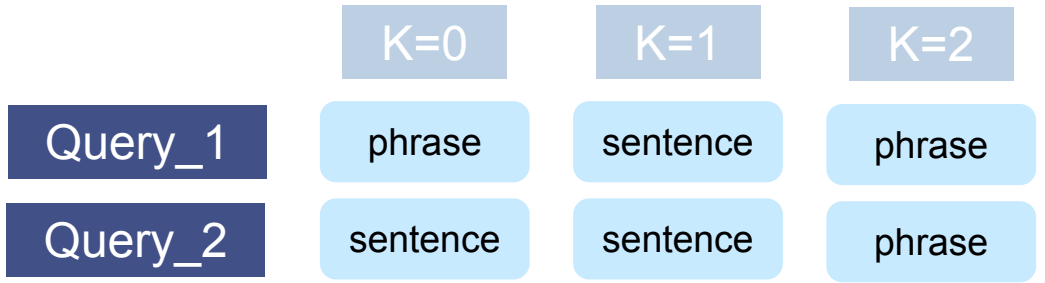
- baseline: sentence만 retrieval
- Multi Unit Retrieval
  - Static : phrase와 sentence의 순서를 고정
  - Dynamic : phrase와 sentence의 순서를 고정하지 않음



<baseline>

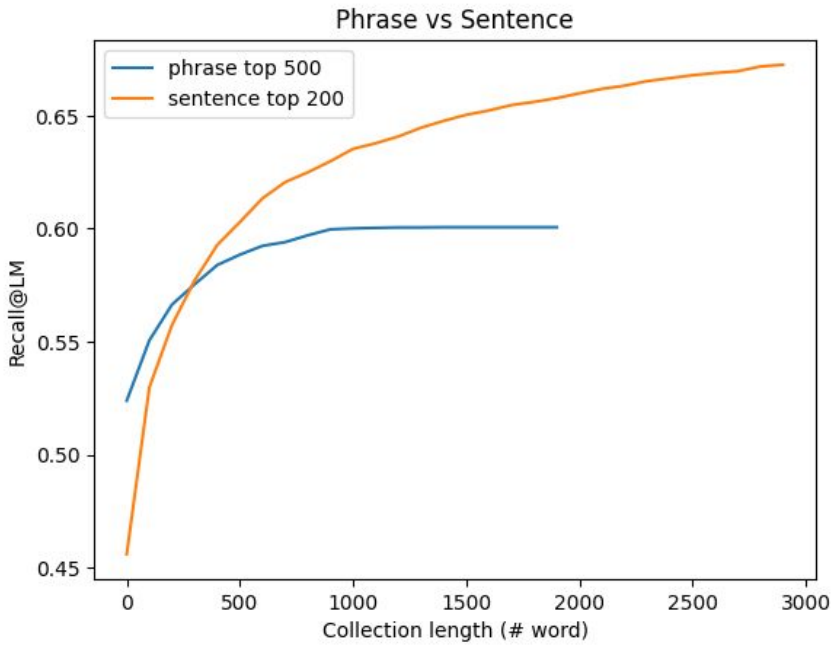


<static>

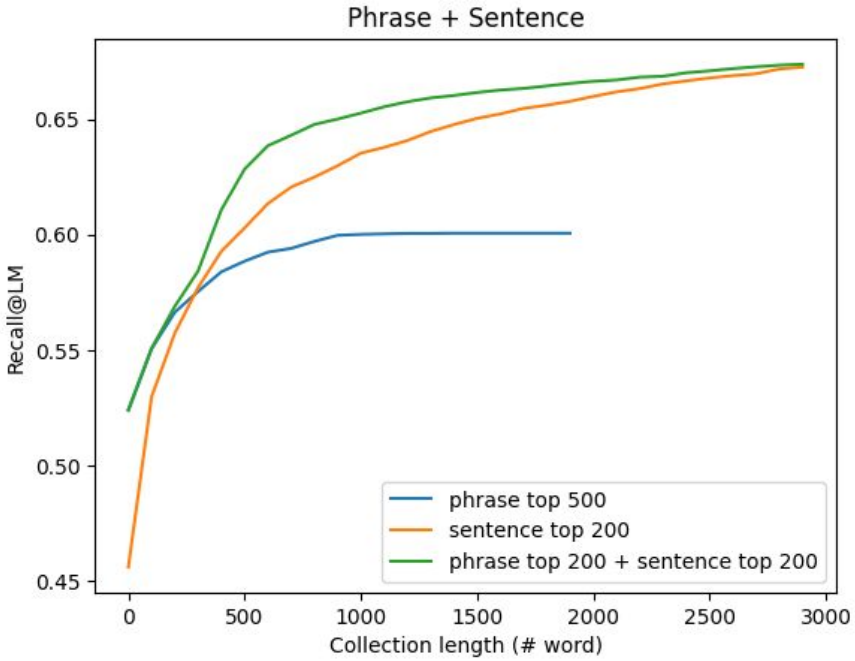


<dynamic>

Multi Unit Retrieval - Static



두 방법의 장점을 모두 활용하기 위해
처음에 phrase를 주고 이어서
sentence를 **주는 방식** 사용



	phrase	sentence
장점	초반에 높은 Recall	높은 Recall 상승폭 높은 최종 Recall
단점	낮은 Recall 상승폭 낮은 최종 Recall	낮은 초반 Recall

	mAR
sentence top 200	0.6316
phrase top 200 + sentence top 200	**0.6450**

Multi Unit Retrieval - Dynamic

- Question과 phrase의 관계에 따라 유동적으로 unit을 결정하고자 **각 unit에 적합한 query encoder**를 찾고자 함.
- Lquery : 정답 phrase가 일치하는지를 구분.
- Lcontext or Ldoc : 정답 phrase뿐만 아니라 정답의 문맥도 같아야 함.

Example

	gold answer	[‘apple’, ‘banana’]	gold context	‘I like apple and banana’
X	phrase	‘apple’	source sentence	‘I ate an apple’
Y	phrase	‘apple’	source sentence	‘I like apple and banana’

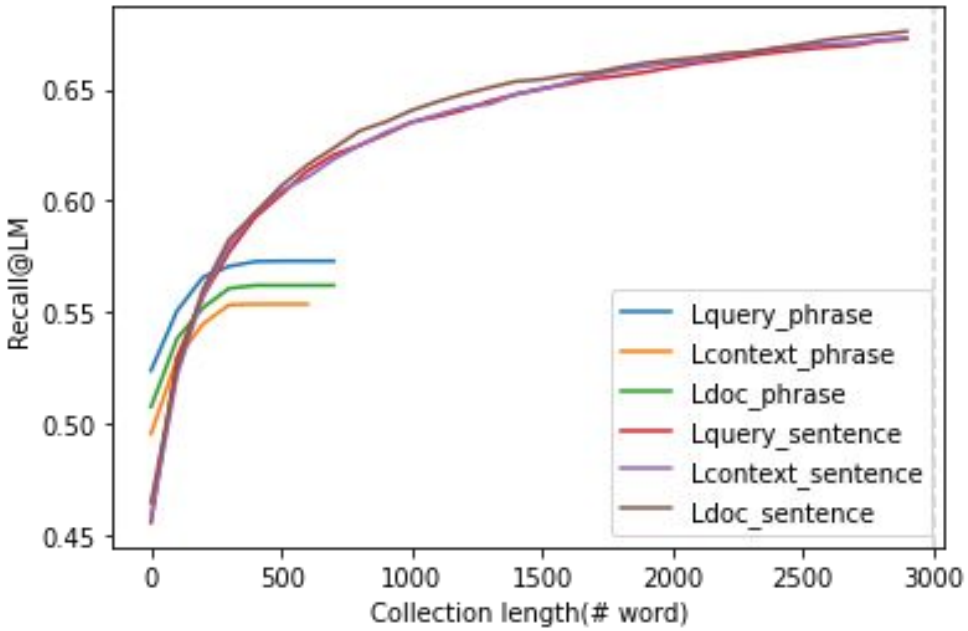
- X는 phrase로 retrieve하는 것이 경제적.
- Y는 sentence로 retrieve하는 것이 recall 면에서 좋음.

- Lquery encoder는 phrase X와 Y를 구분하지 못함.
- Lquery+Lcontext는 X보다 Y를 찾을 경향이 높음.

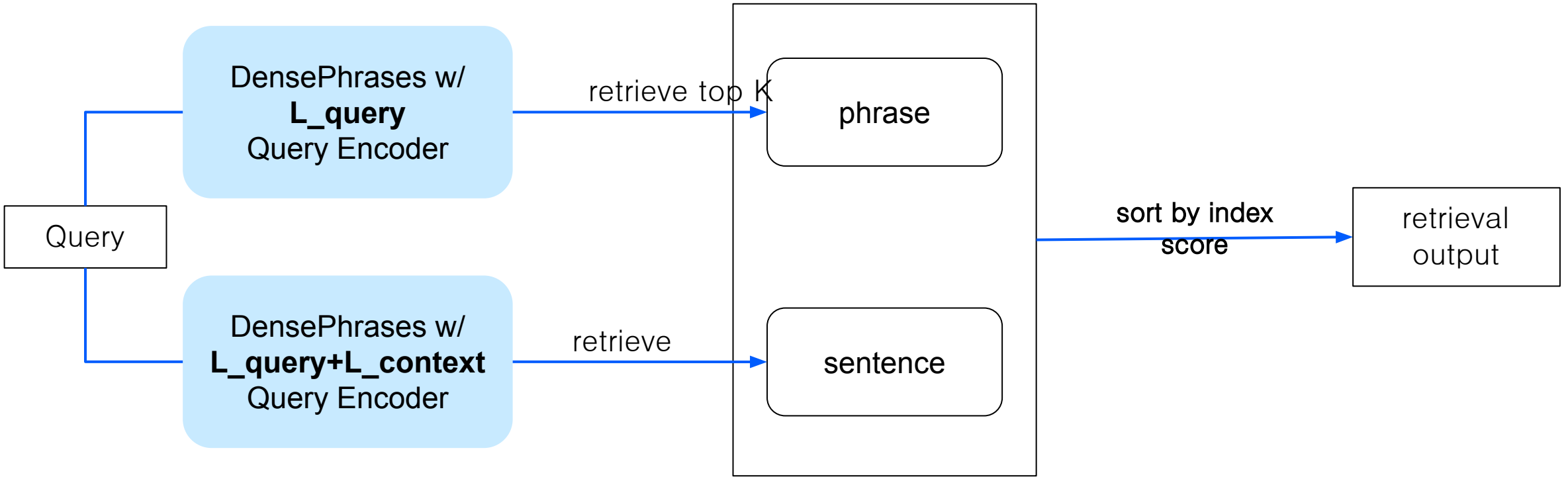
Dual Query Encoder

- Lquery encoder로 retrieve한 phrase, Lquery+Lcontext 또는 Lquery+Lcontext로 retrieve한 sentence 사용

	phrase	sentence
L_query	**0.5629**	0.6316
L_query + L_context	0.5407	0.6322
L_query + L_doc	0.5509	**0.6351**



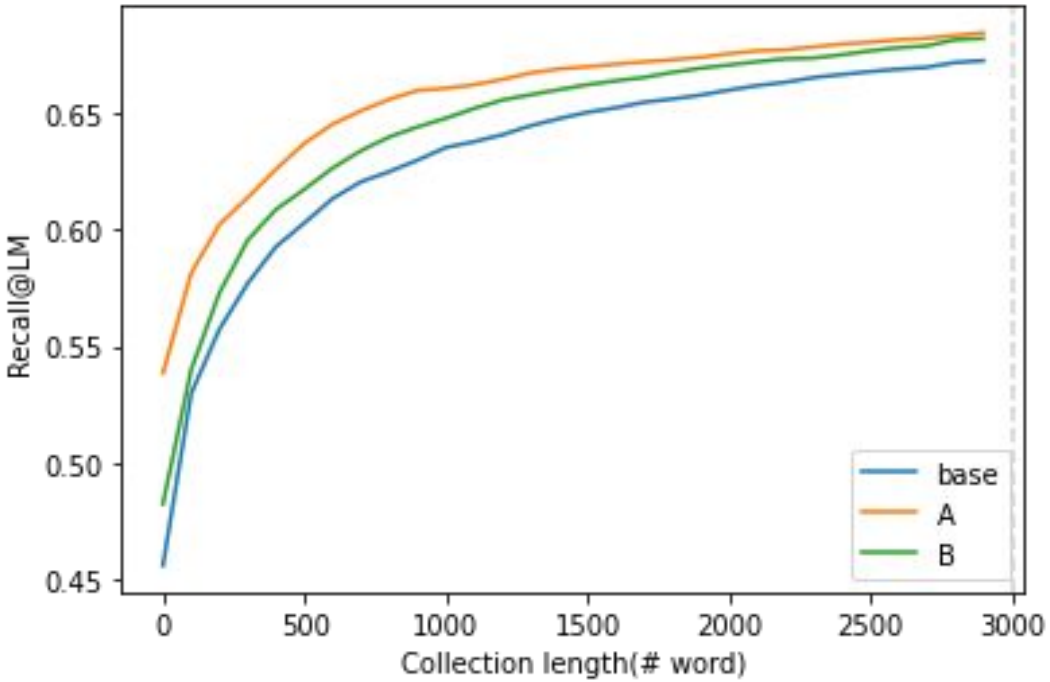
Dynamic Retrieval : Overview



Dual Query Encoder

score : Query vector와 phrase vector의 inner product 값

Multi Unit Retrieval - Dynamic



	Encoder	mAR
base	sentence top 200 : Lquery	0.6316
A	phrase top 200 : Lquery, sentence top 200 : Lquery+Lcontext	**0.6564**
B	phrase top 200 : Lquery, sentence top 200 : Lquery+Ldoc	0.6441

Multi Unit Retrieval - Optimization

- 중복 및 Subset 제거 필요성

  k 번째 반환되는 phrase가 k 번째 이전에 등장했을 경우 최적화 가능

최적화 이전 phrase top 9

top 1 : 'May 18, 2018'

top 2 : 'May 18, 2018'

top 3 : '2", was released in May 2018'

top 4 : '2018'

top 5 : 'February 12, 2016'

top 6 : 'May 2018'

top 7 : 'on May 18, 2018'

top 8 : 'May 18, 2018'

top 9 : '2018'

* 빨간색 : 중복, 파란색 : Subset

최적화 이후 phrase top 5

top 1 : 'May 18, 2018'

top 2 : '2", was released in May 2018'

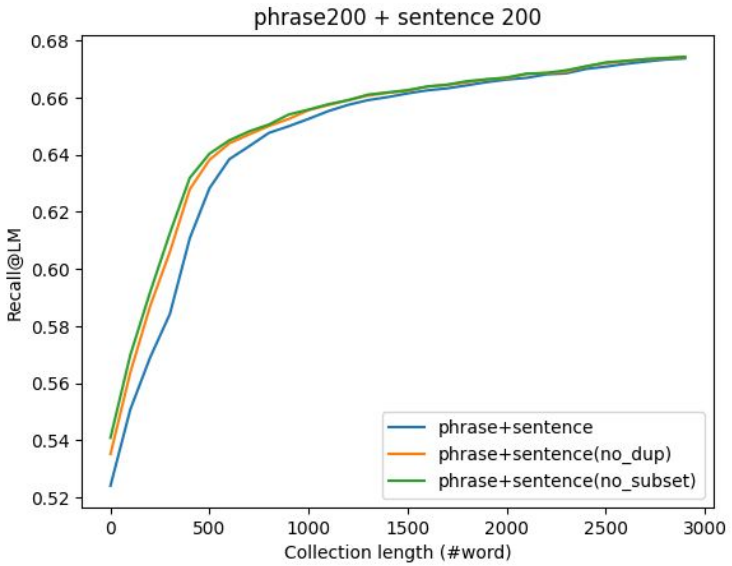
top 3 : 'February 12, 2016'

top 4 : 'on May 18, 2018'

top 5 : 'June 1, 2018'

최적화 이후 9개의 결과가 4개로 줄어든 것을 볼 수 있음

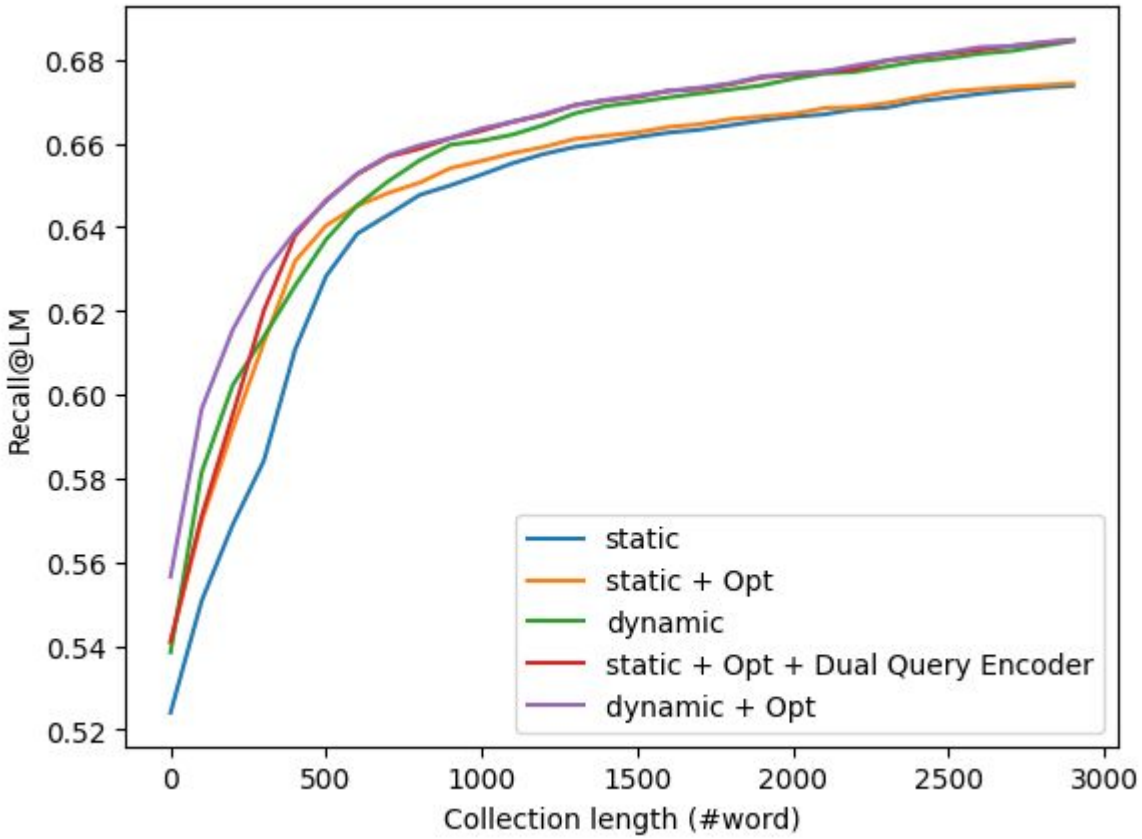
-> **더 적은 # word로 더 다양한 정답을 포함**



	mAR
phrase+sentence	0.6450
phrase+sentence(no_dup)	0.6493
phrase+sentence(no_subset)	**0.6505**

Multi Unit Retrieval : Result

이렇게 4개 어때최적화 한것만 넣어도 될듯? static **0.6173**



	mAR
static	0.6450
static +Opt	0.6505
dynamic	0.6564
static +Opt +Dual	0.6583
dynamic +Opt	**0.6609**

- A : L_query phrase top 200
- B : L_query + L_context sentence top200

4. Conclusion

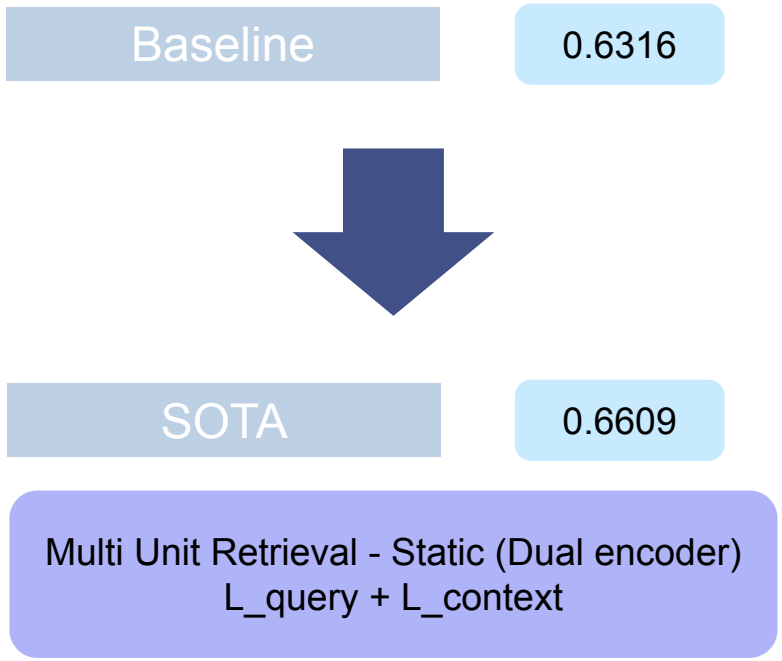
Result

의의 및 한계

추후 발전 계획

4.1. Result

- QSFT
  - Query loss의 단위 변경
    - L_query + L_doc
    - L_query + L_sentence
    - L_query + L_context
  - Loss 구성 요소 추가
  - Knowledge Distillation
- Multi Unit Retrieval
  - Static
    - Single Encoder
    - Dual Encoder
  - Dynamic
    - Dual Encoder
  - 최적화



4.2. 의의 및 한계

다듬기 필요

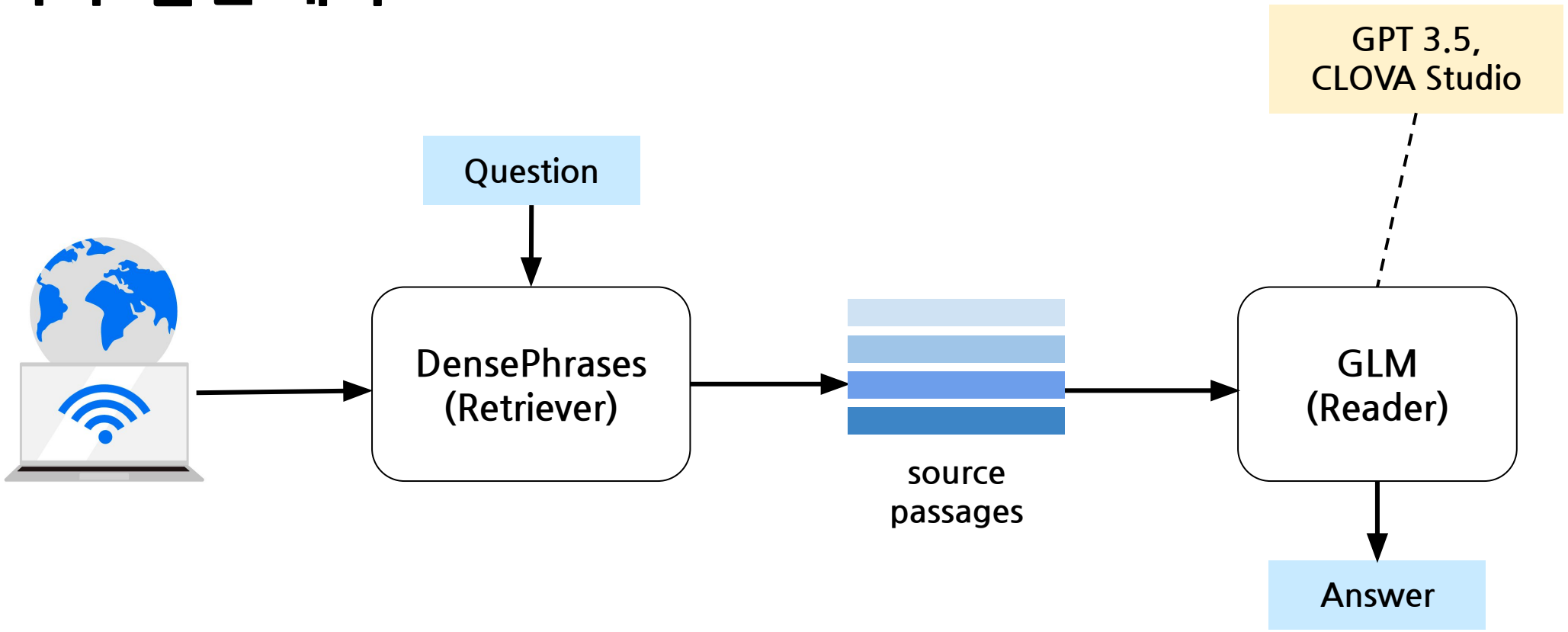
의의

- Retriever에서 recall과 length의 trade-off를 고려하며 **retrieval model의 성능을 향상**시키기 위해 다양한 시도들을 적용.
- 프롬프트의 경량화를 통해 reader model의 **계산 비용을 최소화**하며 **성능은 유지**할 수 있음.

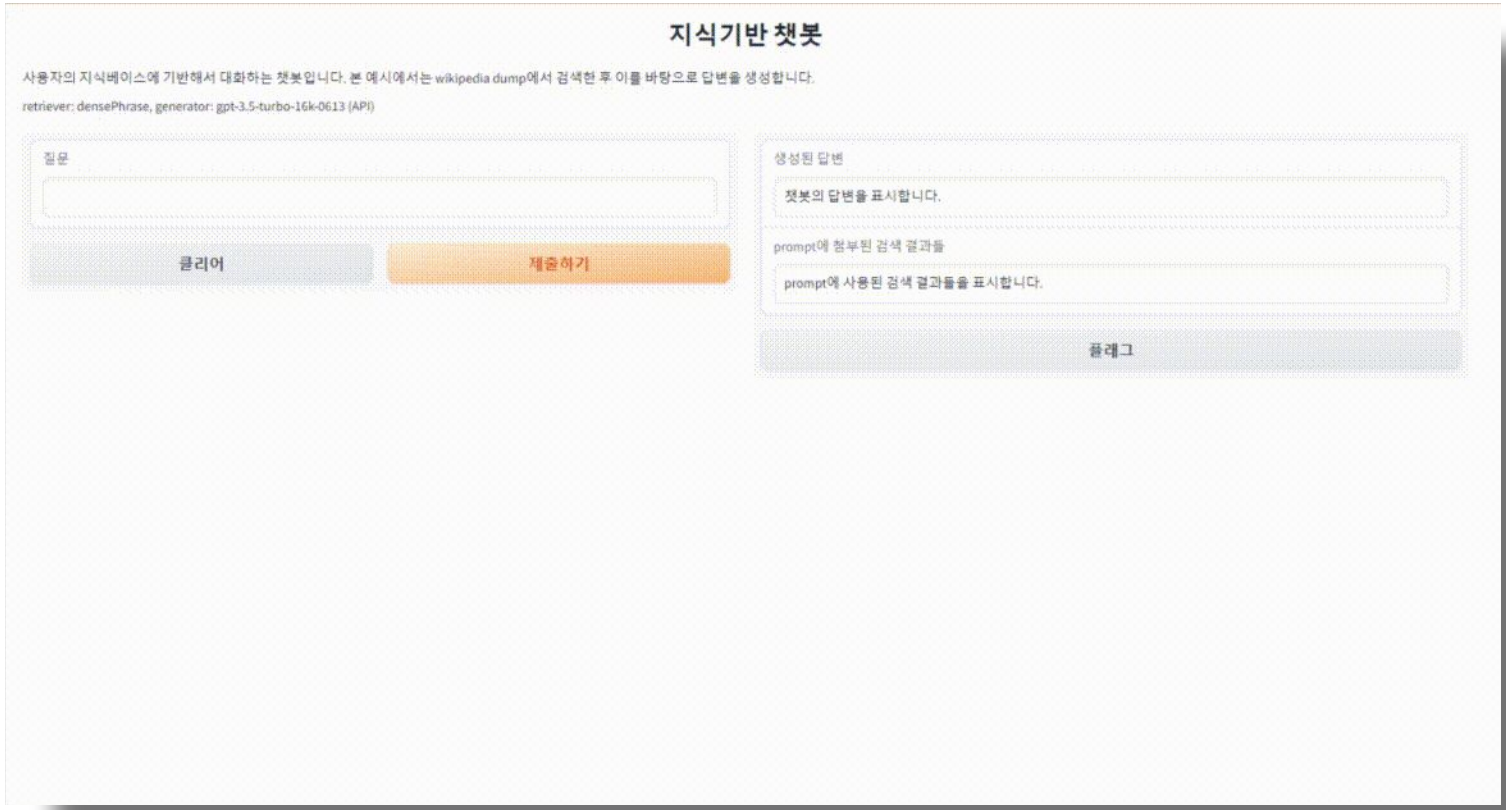
한계

- **실험에 사용된 NQ Dataset의 한계**
  - NQ Dataset은 단답형 질문으로만 구성되어 있고, 질문의 근거가 되는 source document가 하나로 정해져 있음.
  - 그러나, 실제 상황에서는 **1) 단답으로 해결 불가능한 complex query들이 포함될 가능성**이 있고, **2) 질문의 근거가 되는 source document가 여러개 필요**할 수도 있음.
  - 따라서 실제 상황을 반영하는 data는 아니므로, 추후에 이러한 한계점들을 보완하는 여러 data들로 실험을 진행할 필요가 있음.
- **GLM(Generative Language Model)이 받을 수 있는 토큰의 최대 개수**에 따라 성능 차이가 발생할 수 있음.
  - 본 프로젝트는 4096토큰을 사용하는 것을 전제하여 진행하였으나, 대부분의 open source llm들은 2048 tokens까지를 지원하기 때문에 16000tokens까지 사용이 가능한 chatGPT를 제외하면 본 프로젝트의 성능을 최대한으로 발휘할 수 없다.
- Indexing 파일의 크기가 너무 큼 -> **QSFT 시 소요되는 시간이 너무 큼.** (1epoch 당 평균 3시간)
  - 전체 가설을 통합한 결과를 내지 못함.
  - 각 가설들을 더 구체화하지 못함.

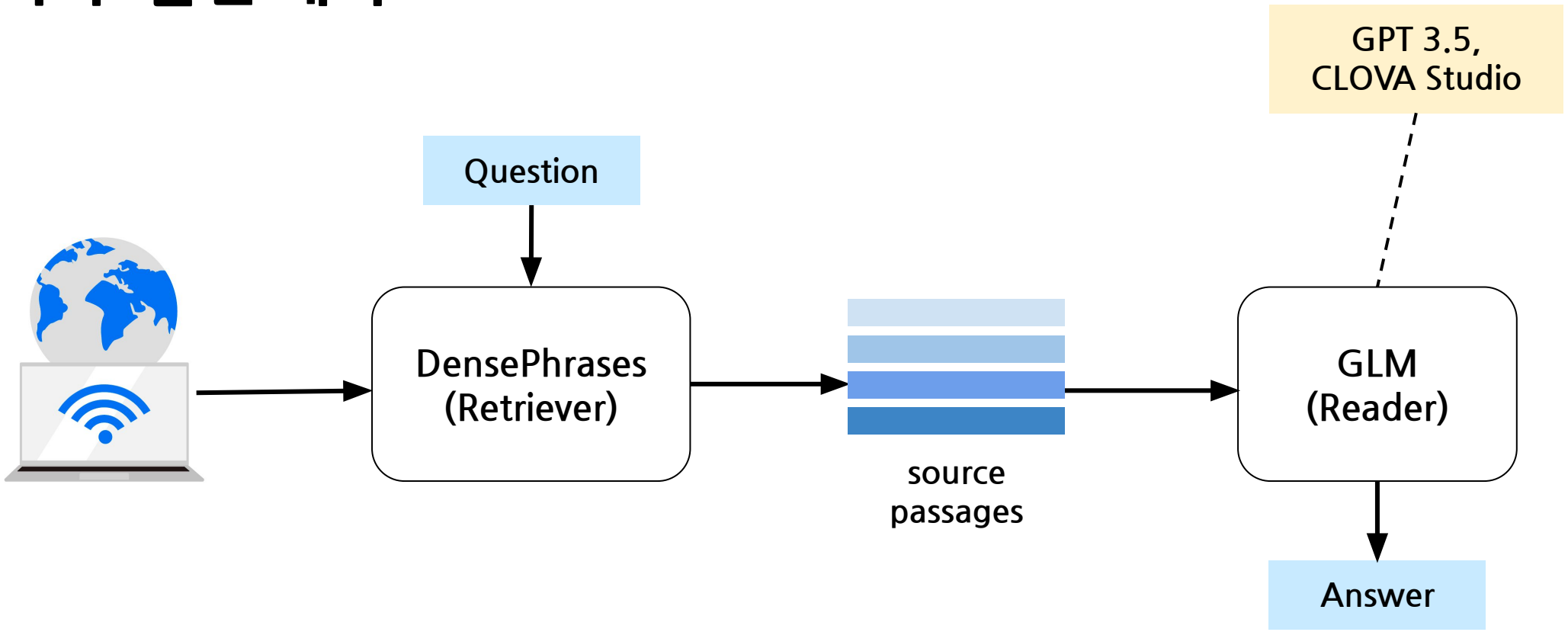
4.3. 추후 발전 계획



지식기반 챗봇 시연



4.3. 추후 발전 계획



5. Reference

5. Reference

1. Jinhyuk Lee, Mujeen Sung, Jaewoo Kang, and Danqi Chen. 2021. Learning Dense Representations of Phrases at Scale. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6634-6647, Online. Association for Computational Linguistics.
2. Jinhyuk Lee, Alexander Wettig, and Danqi Chen. 2021. Phrase Retrieval Learns Passage Retrieval, Too. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3661-3672, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

End of Document
Thank You.