Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, Douwe Kiela

NeurIPS 2020

Abstract

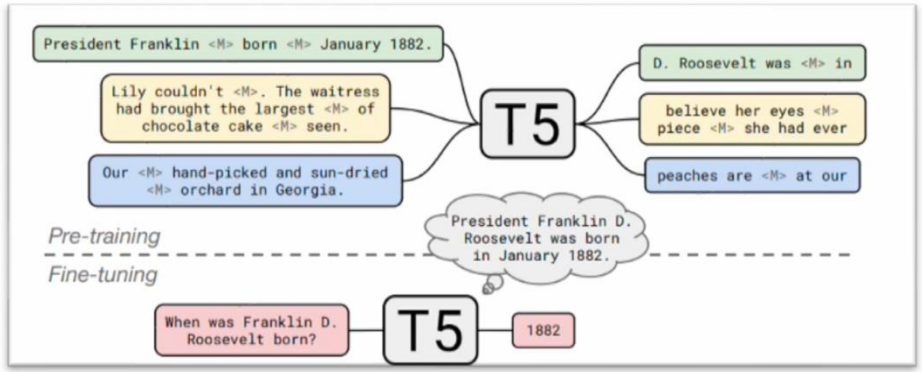
- LLM의 출현으로 medical domain 데이터의 효율적인 처리가 가능해지고, 다양한 작업에서 우수한 성능을 보임

- 13개의 실제 임상/생물의학 NLP task 세트에 대해 Instruction-Finetuned 된 LLM의 성능을 평가

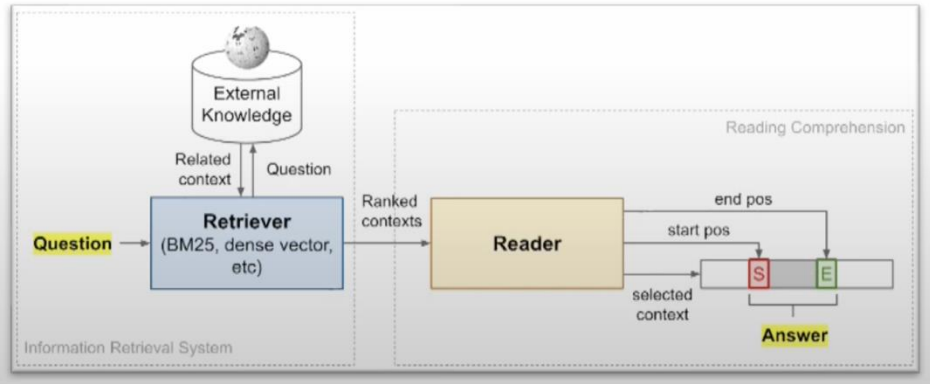
- 결과적으로는 LLM이 대부분의 task에서 의료 전용 모델과 비슷한 성능을 보였고, 특히 QA Task에서 두드러진 성과를 보임

- 하지만 분류/RE(Related Extraction) task에서는 의료 분야 맞춤형 모델인 PubMedBERT 의 성능에는 미치지 못함

• Closed-book QA vs Open-Domain QA



- **Closed-book QA : 질문에 대한 정보를 전혀 주지 않은 채, 모델에게 답변을 생성하도록 하는 태스크**
  - ✓ 외부 지식에 접근할 필요가 없이 end-to-end로 추론 가능
  - ✓ 생성 과정에서 지식을 직접 사용자가 수정 불가
  - ✓ 실제 지식과 관련 없는 hallucination 등이 발생 가능

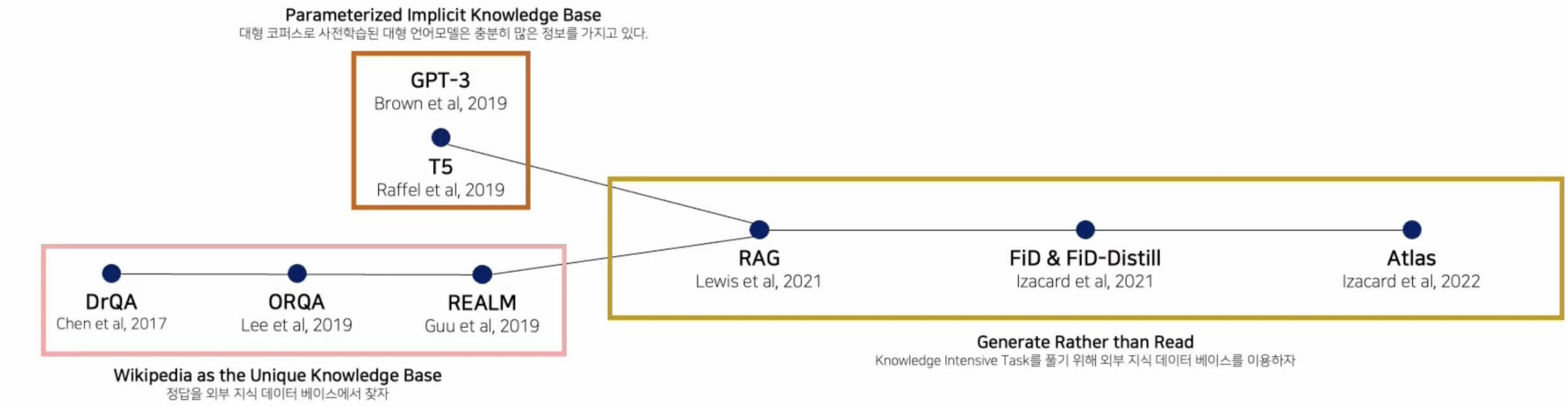


- **Open-Domain QA : 외부 지식에서 답변을 찾는 태스크**
  - ✓ 외부 지식을 직접 사용하기 때문에 지식을 사용자가 수정, 업데이트 가능
  - ✓ 답변을 생성하는 것이 아니기 때문에 답변 내용에 제한이 존재

Background

- Open-Domain Question Answering 태스크의 시작부터 최근 Knowledge Intensive Task까지 각 연구의 흐름
- Knowledge Intensive Task 발전 과정에서 각 논문의 주요 Contribution
  - ➢ RAG는 이전 ODQA 연구를 연결하여 Knowledge Intensive Task로 확장
  - ➢ RAG 이후 연구인 FiD ~ Atlas는 RAG를 기반으로 삼고 있음

Background



Methods

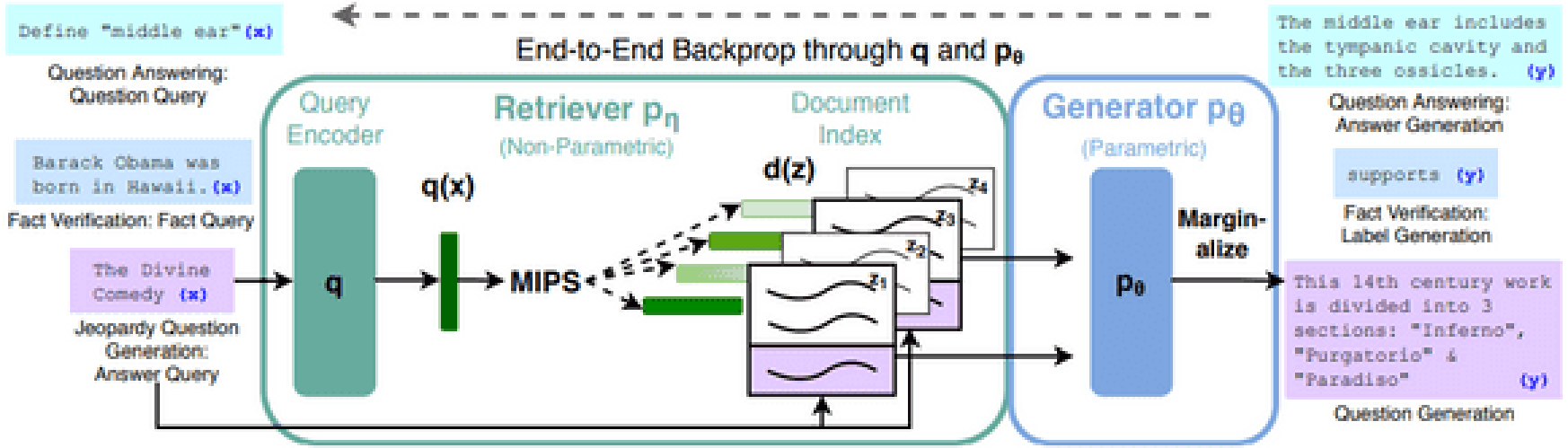


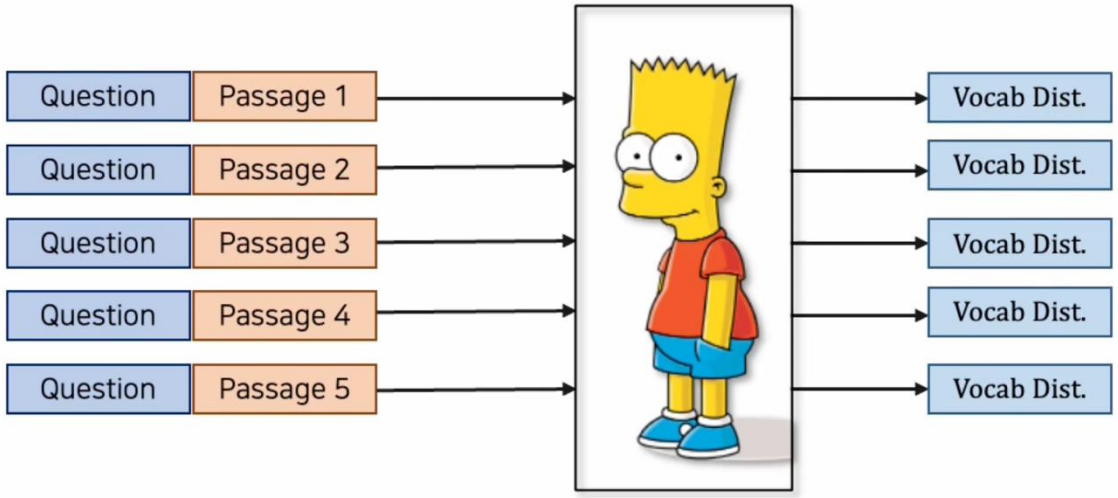
Figure 1: An overview of retrieval-augmented generation (RAG). We combine a pre-trained retriever (**Query Encoder + Document Index**) with a pre-trained encoder-decoder (**Generator**) and fine-tune end-to-end. For some query x, we use Maximum Inner Product Search (MIPS) to find the top-K most relevant documents of all documents z_i. To make the final prediction y, we treat z as a latent variable and marginalize over the encoder-decoder predictions given different documents.

Model

- Retriever : DPR (Dense Passage Retriever)
  - Dense Passage Retrieval for Open-Domain Question Answering
  - EMNLP 2020
  - https://arxiv.org/abs/2004.04906

Model

- Generator : BART
  - BART의 인코더, 디코더를 사용하여 모델링
  - 400M parameter를 가진 BART-large 사용

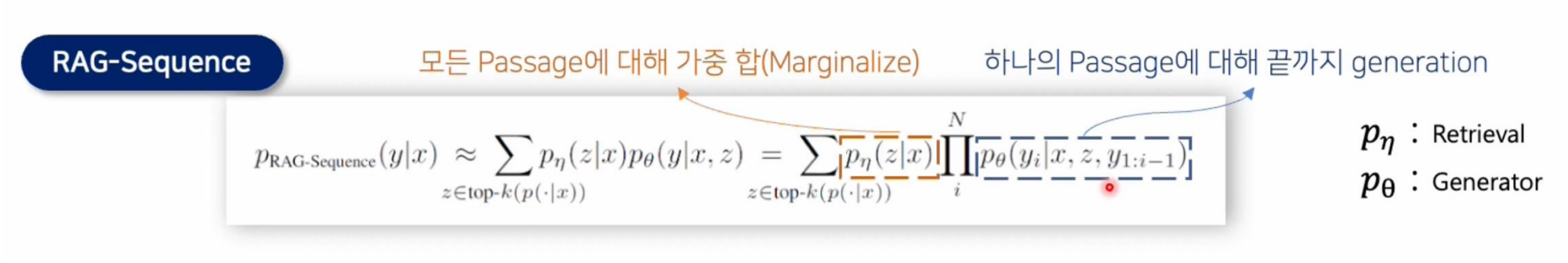


- **Query와 가까운 Top-5 Passage를 이용하여 BART Input 구성**
- **각 Passage와 Query를 Concat하여 BART Input 생성**
  - ✓ 하나의 Query 당 5개의 BART Input 존재 → 배치 처리를 통해 추론
- **각 (Passage, Question) 에서 Vocab Distribution 생성**
  - ✓ Aggregation 과정을 통해 하나의 문장 생성

How to decode?

- RAG-Sequence Model
  - 각 passage마다 별개로 생성
  - 최종 생성문을 시점 단위로 marginalize

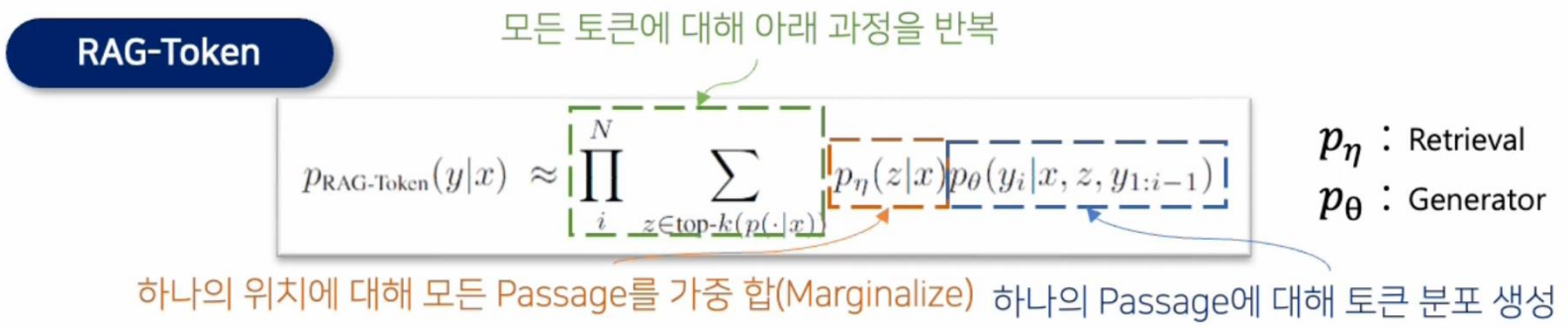
  - RAG-Sequence Model은 동일하게 검색된 문서를 사용하여 target sequence를 생성한다. top-K approximation을 통해 seq2seq probability p(x|y)p(x|y)를 얻기위해 검색된 passage를 marginalized single latent variable로 처리



How to decode?

- RAG-Token Model
  - 토큰 생성마다 passage별 분포 marginalize
  - 다음 토큰 생성 시 이전 marginalize 결과를 이용

  - RAG-Token Model은 각 target token에 대해 다른 latent passage를 사용한다. 이를 통해 generator는 답변을 생성할 때 여러 document에서 내용을 선택할 수 있음



Experiments

- Open-domain Question Answering

- 총 4가지의 Open-domain QA Dataset 이용
  1. Natural Questions(NQ)
  2. TriviaQA(TQA)
  3. WebQuestions(WQ)
  4. CuratedTrec(CT)

	Model	NQ	TQA	WQ	CT
Closed-Book	T5-11B [46]	34.5	- /50.1	37.4	-
	T5-11B + SSM [46]	36.6	- /60.5	44.7	-
Open-Book	REALM [18]	40.4	- / -	40.7	46.8
	DPR [22]	41.5	**57.9**/ -	41.1	50.6
	RAG-Token	44.1	55.2/66.1	**45.5**	50.0
	RAG-Sequence	**44.5**	56.1/**68.0**	45.2	**52.2**

Table 1: Open-Domain QA Test Scores. For TQA, the left column uses the test split commonly used in Open-Domain QA. The right column uses the hidden TQA Wiki test split. See Appendix B for further information.

Experiments

- 기존의 QA 태스크와 다르게 Generation 으로 접근하였음
  - 이는 단순히 문서에서 원하는 내용을 찾고 끝나는 것이 아니라, 문장을 생성하는 Generator 성능이 월등히 올라간 시점에서 이를 잘 활용하였다고 생각함

- Reader를 활용한 DPR보다 좋은 성능
- 외부 지식을 사용하지 않은 Closed-Book보다 월등히 좋은 성능

	Model	NQ	TQA	WQ	CT
Closed-Book	T5-11B [46]	34.5	- /50.1	37.4	-
	T5-11B + SSM [46]	36.6	- /60.5	44.7	-
Open-Book	REALM [18]	40.4	- / -	40.7	46.8
	DPR [22]	41.5	**57.9**/ -	41.1	50.6
	RAG-Token	44.1	55.2/66.1	**45.5**	50.0
	RAG-Sequence	**44.5**	56.1/**68.0**	45.2	**52.2**

Table 1: Open-Domain QA Test Scores. For TQA, the left column uses the test split commonly used in Open-Domain QA. The right column uses the hidden TQA Wiki test split. See Appendix B for further information.

Ablation Study

- RAG의 Retriever를 BM25(TF-IDF Based) 로 교체
  - FEVER에서는 좋은 성능 : FEVER는 사실 확인 task로 실제 문서에서 중요 토큰의 등장 여부가 가장 중요 (의미적 정보는 중요 x)
- Retriever Freeze : Generator만 학습
  - Retriever 역시 학습하는 것이 효과적임.

Model	NQ	TQA	WQ	CT	Jeopardy-QGen		MSMarco		FVR-3	FVR-2
		Exact Match			B-1	QB-1	R-L	B-1	Label Accuracy	
RAG-Token-BM25	29.7	41.5	32.1	33.1	17.5	22.3	55.5	48.4	**75.1**	**91.6**
RAG-Seq-BM25	31.8	44.1	36.6	33.8	11.1	19.5	56.5	46.9		
RAG-Token-Frozen	37.8	50.1	37.1	51.1	16.7	21.7	55.9	49.4	72.9	89.4
RAG-Seq-Frozen	41.2	52.1	41.8	52.6	11.8	19.6	56.7	47.3		
RAG-Token	43.5	54.8	**46.5**	51.9	**17.9**	**22.6**	56.2	**49.4**	74.5	90.6
RAG-Seq	**44.0**	**55.8**	44.9	**53.4**	15.3	21.5	**57.2**	47.5		

Table 5: Ablations on the development set. As FEVER is a classification dataset, RAG-Token and RAG-Sequence are equivalent.

Thank you