KOREA UNIVERSITY

DSBA Open QA Study

Open-Domain Question Answering Paper Review #4

DSBA
Data Science & Business Analytics

고려대학교 산업경영공학과

Data Science & Business Analytics Lab

발표자 : 이유경

00

발표 목차

Contents

01 Introduction

논문 소개

Part 5 : Dense Retriever and End-to-end Training

REALM: Retrieval-Augmented Language Model Pre-Training.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, Ming-Wei Chang. ICML 2020.

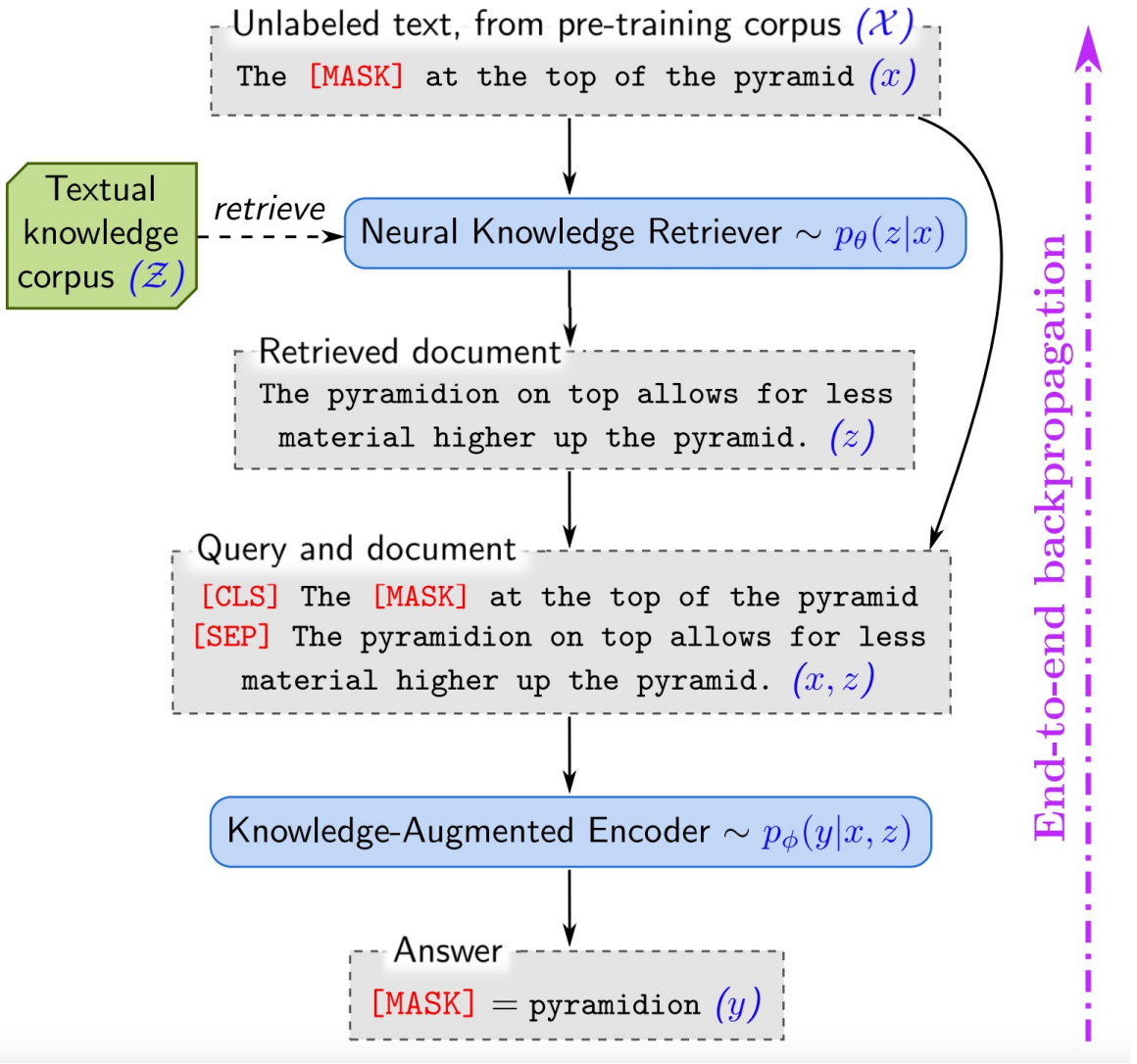
Part 7 : Open-Domain Question Answering using Text & Knowledge Bases

Knowledge Guided Text Retrieval and Reading for Open Domain Question Answering.

Sewon Min, Danqi Chen, Luke Zettlemoyer, Hannaneh Hajishirzi. arXiv 2019.

02 REALM: Retrieval-Augmented Language Model Pre-Training.

Overview



- TL;DR
  - Retriever도 학습하면 QA 성능이 매우 높아짐
  - Retriever 와 reader를 한번에 학습(Joint training)할 수 있음
  - Retriever를 Pretraining에서 수행하는 모델 제안

- Main Contribution
  - Retriever와 Reader를 한번에 학습하는 E2E 모델
  - Query을 넣어(input), 답(output)을 찾는 과정을 두 단계로 분리
    - Neural Knowledge Retriever
      - Query -> Query 의 답이 될만한 document를 찾음
    - Knowledge-Augmented Encoder
      - Retrieved Document -> Answer
  - Pretraining과 Finetuning(ODQA)을 모두 진행함

02 REALM: Retrieval-Augmented Language Model Pre-Training.

Background

1. **Pretrained LM의 능력과 한계**
   - PLM은 Pretrain 단계에서 이미 large corpora로 학습되므로 대량의 정보를 포함하고 있음
   - 대부분의 PLM은 Cloze task로 학습을 진행하기 때문에 Mask를 예측하는 과정에서 언어를 이해할 뿐만 아니라 정보를 습득함

     :"The [Mask] is the currency of the United Kingdom" (answer:"pound").
   - 하지만, PLM이 정보를 저장하는 방식은 **"implicitly"** 함
     - Network에 어떤 knowledge가 학습되어 있는지 알 수 없음
     - 더 많은 knowledge를 학습하기 위해서는 model size를 증가 시켜야 하며, 계산 비용이 상당함

2. **Explicit하게 Knowledge를 학습 및 저장하는 모델 필요**
   - Textual knowledge retriever를 통해 기존 PLM을 보다 해석 가능하고 explicit하게 knowledge를 학습하는 모델로 개선
   - 즉, Retriever 과정이 pretraining에 포함되어있는 형태임
   - 문장 -> Retriever -> 정답을 찾아낼 수 있는 새로운 모델 구조를 제안함

02 REALM: Retrieval-Augmented Language Model Pre-Training.

Method

- Main Idea
  - Original QA : Query(x)를 넣어 Answer(y)를 찾겠어
  - REAML
    - Step 1 : Query(x)를 넣어 Retrieved document(z)를 찾고
    - Step 2 : Query(x)와 Retrieved document(z)를 넣어 Answer(y)를 찾겠어

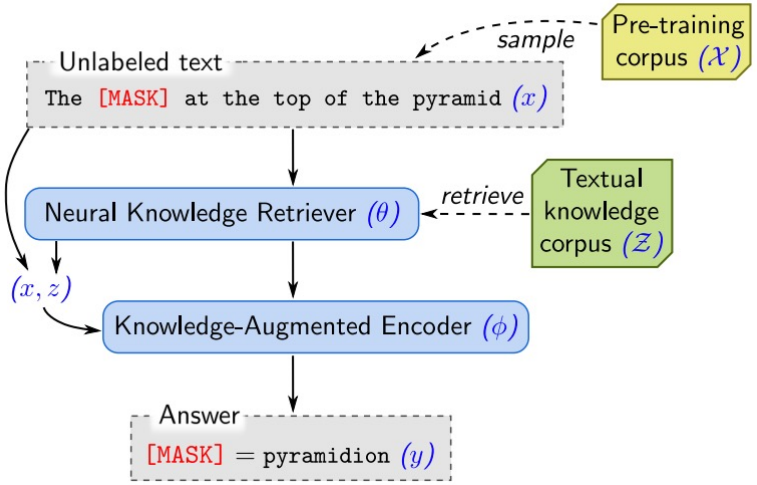
$$p(y|x) = \sum_{z \in \mathcal{Z}} p(y|z,x)\, p(z|x).$$

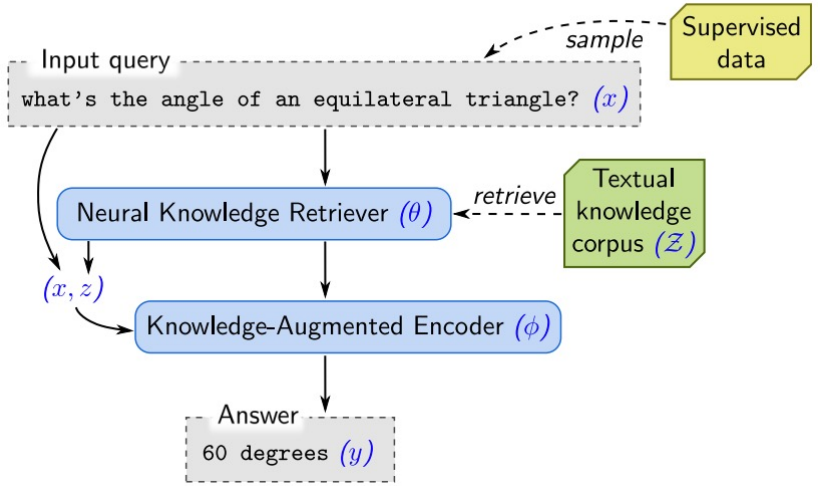
Step 2 Step 1

All knowledge corpus

- Model Architecture
  - Neural knowledge retriever (Step 1)
  - knowledge-augmented encoder (Step 2)

- Training Process
  - Unsupervised (Pretraining)
  - Supervised training (Finetuning)
    - QA Task





02 REALM: Retrieval-Augmented Language Model Pre-Training.

Method

(시간 관계상) 발표에서 제외했지만
Top K를 approximate 하는 과정에서
Maximum Inner Product Search 라는
방법론을 자세히 설명하고 있음

- Neural Knowledge Retriever
  - Retriever는 dense inner product model로 정의됨 == "Query와 relevance score가 가장 높은 document를 찾겠다"는 의미

$$p(y|x) = \sum_{z \in \mathcal{Z}} \underbrace{p(y|x,z)}_{\text{reader}} \underbrace{p(z|x)}_{\text{retriever}} \approx \sum_{z \in \text{TOP}_k(\mathcal{Z})} p(y|x,z)p(z|x)$$

① Distribution, relevance score

$$p(z|x) = \frac{exp\ f(x,z)}{\sum_{z'} exp\ f(x,z')},$$
$$f(x,z) = \text{Embed}_{\text{input}}(x)^{\top} \text{Embed}_{\text{doc}}(z),$$

② BERT style transformer

$$\text{join}_{\text{BERT}}(x) = [\text{CLS}]x[\text{SEP}]$$
$$\text{join}_{\text{BERT}}(x_1, x_2) = [\text{CLS}]x_1[\text{SEP}]x_2[\text{SEP}]$$

③ Embeddings

$$\text{Embed}_{\text{input}}(x) = \mathbf{W}_{\text{input}} \text{BERT}_{\text{CLS}}(\text{join}_{\text{BERT}}(x))$$
$$\text{Embed}_{\text{doc}}(z) = \mathbf{W}_{\text{doc}} \text{BERT}_{\text{CLS}}(\text{join}_{\text{BERT}}(z_{\text{title}}, z_{\text{body}}))$$

- Knowledge-Augmented Encoder
  - Retrieved documents와 query를 함께 사용하여 정답을 찾아내며, pretraining과 finetuning 과정이 살짝 다름

① Pretraining (MLM)

$$p(y|z,x) = \prod_{j=1}^{J_x} p(y_j|z,x)$$
$$p(y_j|z,x) \propto exp\left(w_j^{\top} \text{BERT}_{\text{MASK}}(j)(\text{join}_{\text{BERT}}(x, z_{\text{body}}))\right)$$

: Mask 예측

② Open QA Finetuning

$$p(y|z,x) \propto \sum_{s \in S(z,y)} \exp\left(\text{MLP}\left(\left[h_{\text{START}(s)}; h_{\text{END}(s)}\right]\right)\right)$$
$$h_{\text{START}(s)} = \text{BERT}_{\text{START}}(s)(\text{join}_{\text{BERT}}(x, z_{\text{body}})),$$
$$h_{\text{END}(s)} = \text{BERT}_{\text{END}}(s)(\text{join}_{\text{BERT}}(x, z_{\text{body}})),$$

: Span 예측

02 REALM: Retrieval-Augmented Language Model Pre-Training.

Method

- Injecting inductive biases into pre-training
  - (개인적 견해) 타 연구들은 비슷한 방법론을 사용할 때 infuse knowledge 라고 표현하는 경우가 많은데,
    해당 연구는 injecting inductive biases라는 표현 사용함 – 논문 쓸 때 참고해봐도 좋을 포인트라 생각
  - Salient span masking
    - Named entity를 사용하여 salient spans 생성 (such as “United Kingdom” or “July 1969”)
    - 해당 masking strategy의 목적은 정답일 가능성이 높은 Entity 들을 Query의 Answer로 간주한 것
    - Pretraining REALM 과정에서 [MASK] 자리에 들어올 값을 Top K Retriever 과정을 통해 찾아내는것이 포인트
    - Span masking은 대부분 비슷한 방법론을 취하고 있음 (SpanBERT, MASS, BART, Pegasus 등)
    - BERT masking , SpanBERT masking, REALM masking을 비교한 결과 제안 방법이 가장 좋았다고 언급함

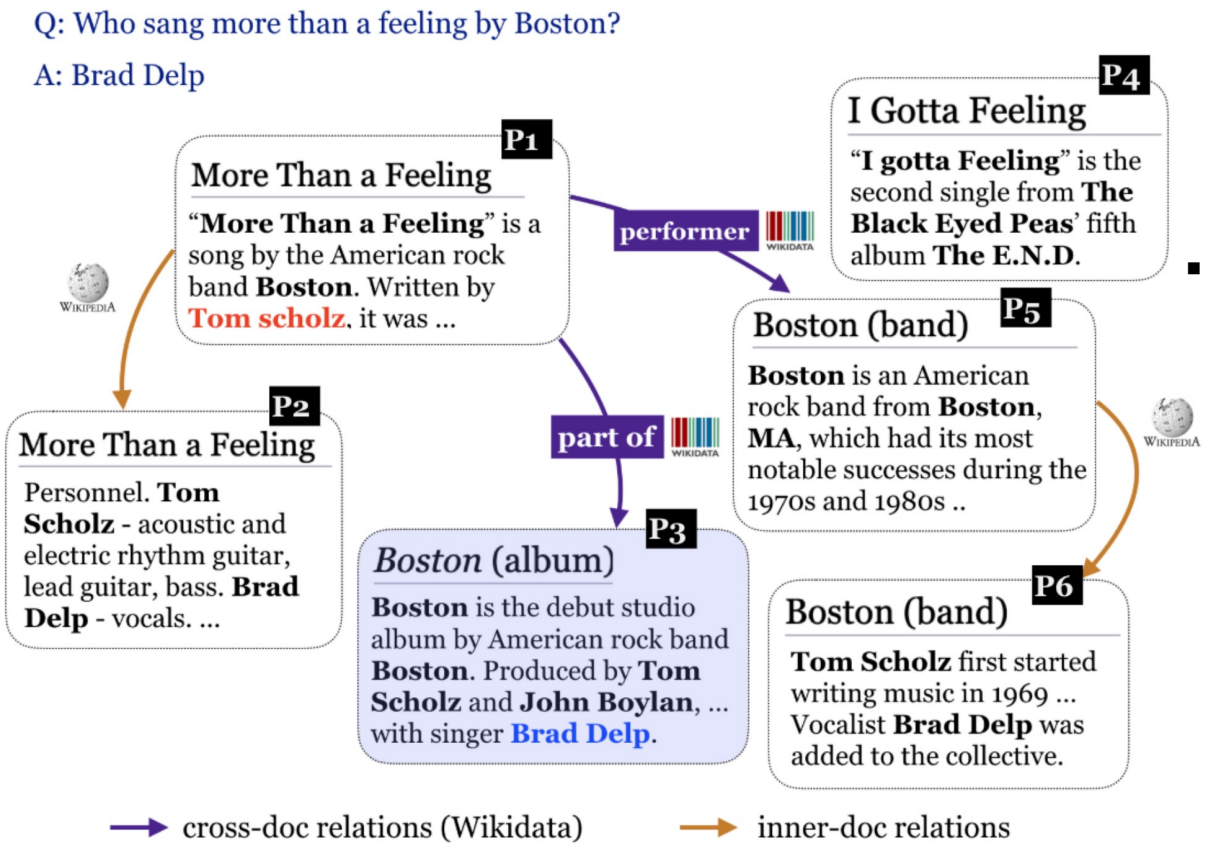
02 REALM: Retrieval-Augmented Language Model Pre-Training.

Result

Name	Architectures	Pre-training	NQ (79k/4k)	WQ (3k/2k)	CT (1k /1k)	# params
BERT-Baseline (Lee et al., 2019)	Sparse Retr.+Transformer	BERT	26.5	17.7	21.3	110m
T5 (base) (Roberts et al., 2020)	Transformer Seq2Seq	T5 (Multitask)	27.0	29.1	-	223m
T5 (large) (Roberts et al., 2020)	Transformer Seq2Seq	T5 (Multitask)	29.8	32.2	-	738m
T5 (11b) (Roberts et al., 2020)	Transformer Seq2Seq	T5 (Multitask)	34.5	37.4	-	11318m
DrQA (Chen et al., 2017)	Sparse Retr.+DocReader	N/A	-	20.7	25.7	34m
HardEM (Min et al., 2019a)	Sparse Retr.+Transformer	BERT	28.1	-	-	110m
GraphRetriever (Min et al., 2019b)	GraphRetriever+Transformer	BERT	31.8	31.6	-	110m
PathRetriever (Asai et al., 2019)	PathRetriever+Transformer	MLM	32.6	-	-	110m
ORQA (Lee et al., 2019)	Dense Retr.+Transformer	ICT+BERT	33.3	36.4	30.1	330m
Ours ($\mathcal{X}$ = Wikipedia, $\mathcal{Z}$ = Wikipedia)	Dense Retr.+Transformer	REALM	39.2	40.2	**46.8**	330m
Ours ($\mathcal{X}$ = CC-News, $\mathcal{Z}$ = Wikipedia)	Dense Retr.+Transformer	REALM	**40.4**	**40.7**	42.9	330m

03 Graph Retriever

Overview



- TL;DR
  - ODQA 할 때 KB를 사용해서 Question에서 정보를 추출
  - Wikipedia passage로 knowledge graph 생성
  - GCN – Span prediction 진행
- Main Contribution
  - External information(KB)을 충분히 활용한 연구
  - WIKIDATA의 relation을 활용하여 KG 생성
  - 다양한 관계를 고려한 Graph를 기반으로 GCN modeling
  - 모델은 크게 Graph Retriever와 Graph Reader로 구성
  - Graph Retriever
    - Entity linking, TF-IDF (Question)
    - 관련된 article retrieval
  - Graph Reader
    - Bert – passage, relation encoding (respectively)
    - Encoded representation – GCN – New representation

03 Graph Retriever

Background

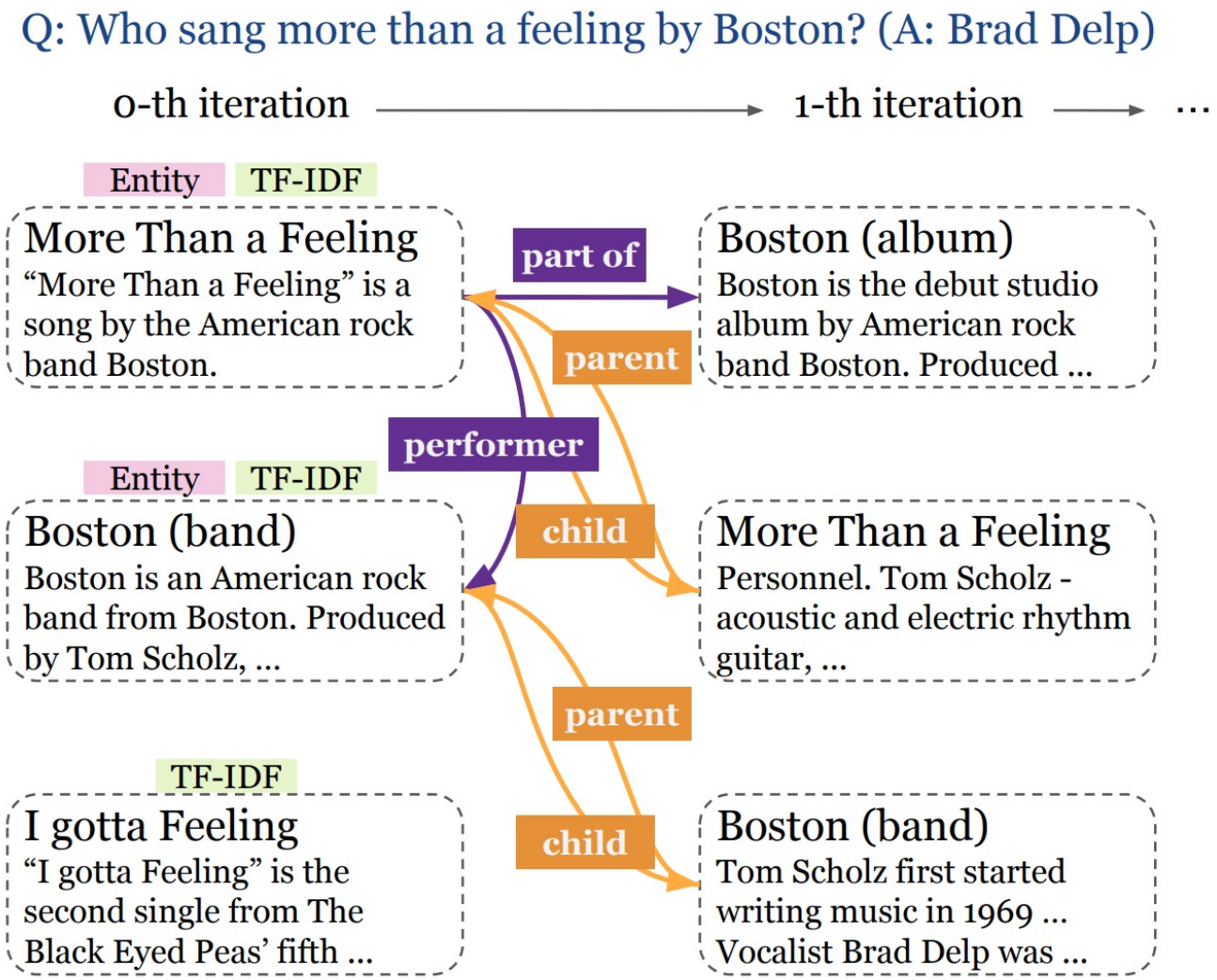
1. **Text-based QA의 한계**
   - Text-based QA는 Retriever – Reader 구조로 진행
   - Retrieval 과정에서 사용 가능한 데이터가 제한 되어있다는 한계점이 존재
   - Passage와 Question 사이의 관계들을 고려하지 못한다는 한계점 존재

2. **Relation을 고려하는 새로운 QA 모델의 필요성**
   - 기존에도 External knowledge를 사용하여 모델링 하는 경우는 존재하였음 (KB, 다양한 외부 Source 사용)
   - WIKIDATA는 각 Entity별 다양한 relation이 정의도어있음
   - WIKIDATA의 특성을 고려하여 각 정보들간 Relation을 고려한 Graph를 구성하고 의미있는 passage를 찾을 수 있음
   - Passage를 collect한 후 information을 fuse 할 수 있으며 relation으로 부터 파생된 정보를 model이 학습할 수 있게 됨

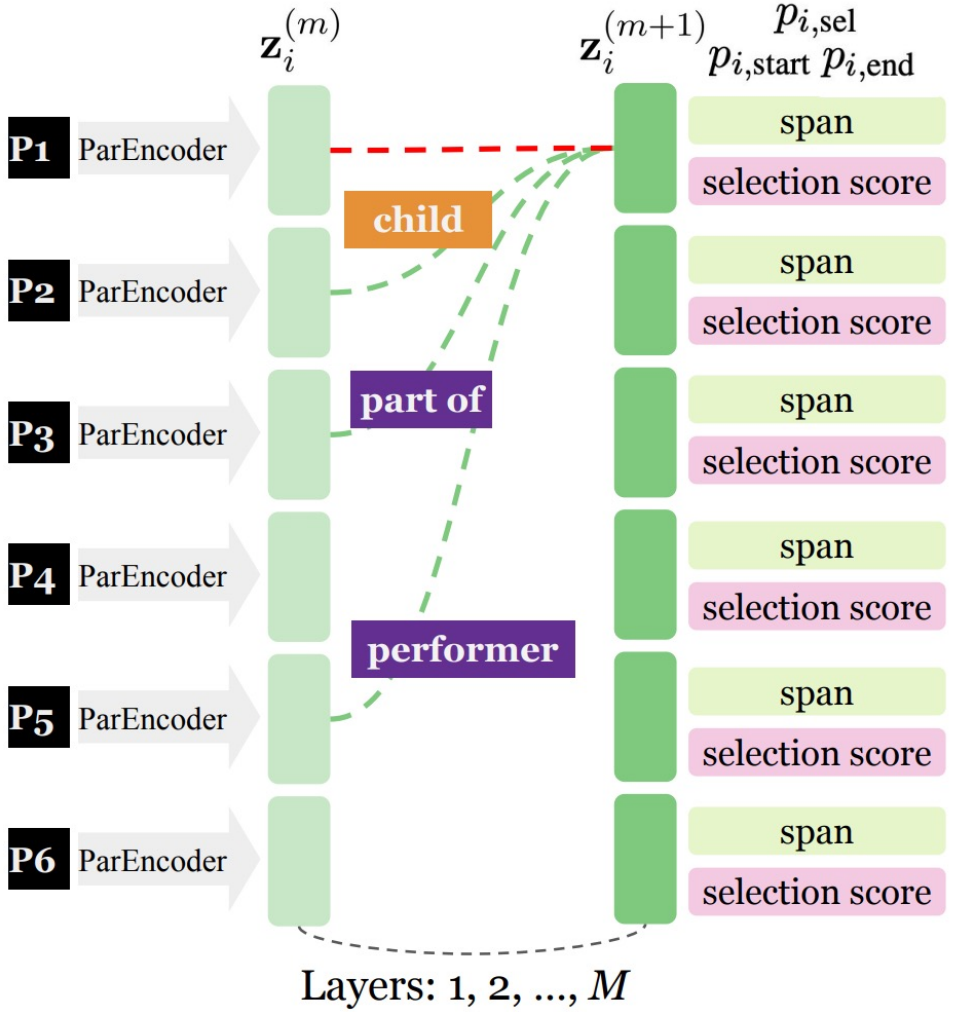
Subject	Relation	Object
More Than a Feeling	performer	Boston (band)
More Than a Feeling	part of	Boston (album)
More Than a Feeling	genre	Hard rock
More Than a Feeling	country of origin	USA
More Than a Feeling	record label	Epic
More Than a Feeling	followed by	Foreplay/Long Time

03 Graph Retriever

Method



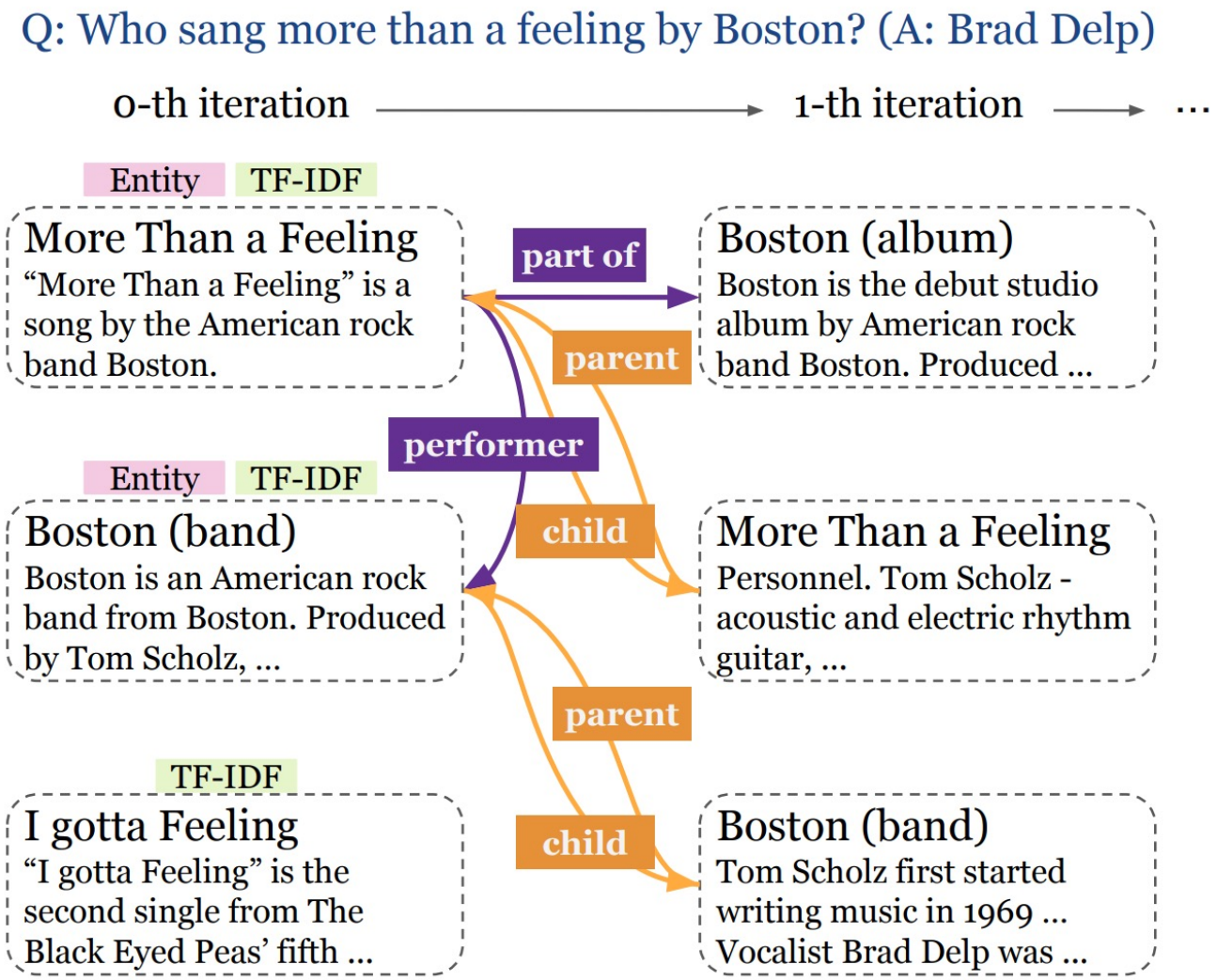
Graph Retriever



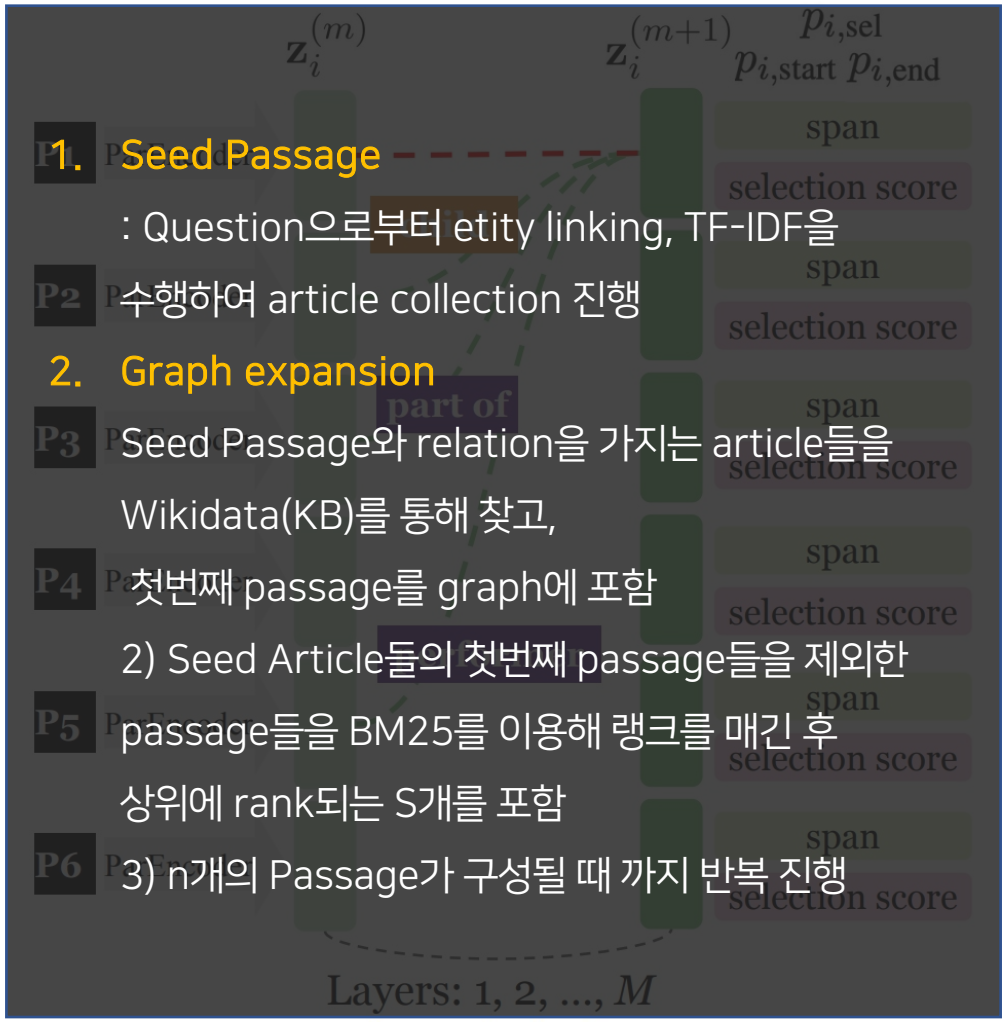
Graph Reader

03 Graph Retriever

Method



Graph Retriever

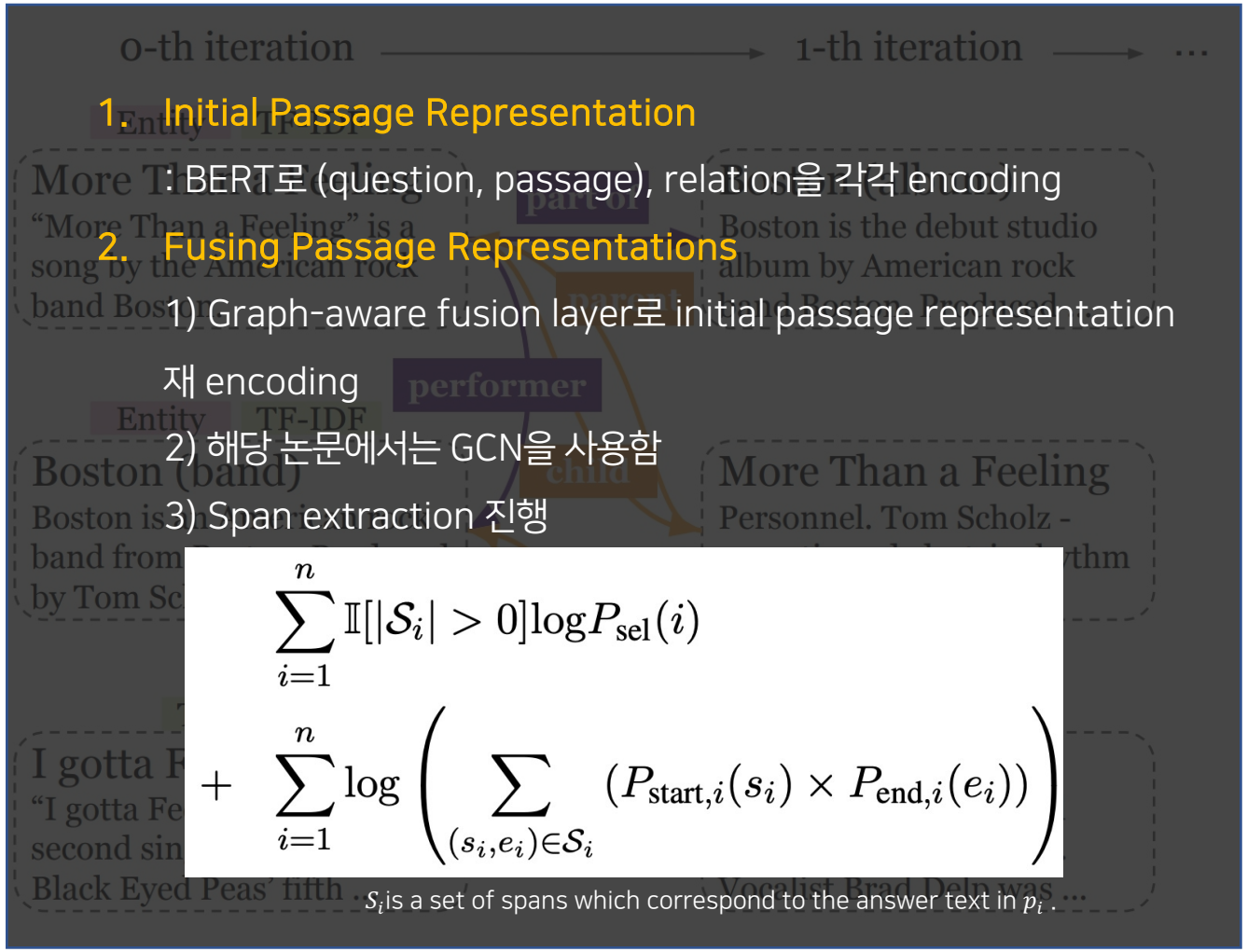


Graph Reader

03 Graph Retriever

Method

Q: Who sang more than a feeling by Boston? (A: Brad Delp)



1. Initial Passage Representation

: BERT로 (question, passage), relation을 각각 encoding

2. Fusing Passage Representations

1) Graph-aware fusion layer로 initial passage representation 재 encoding

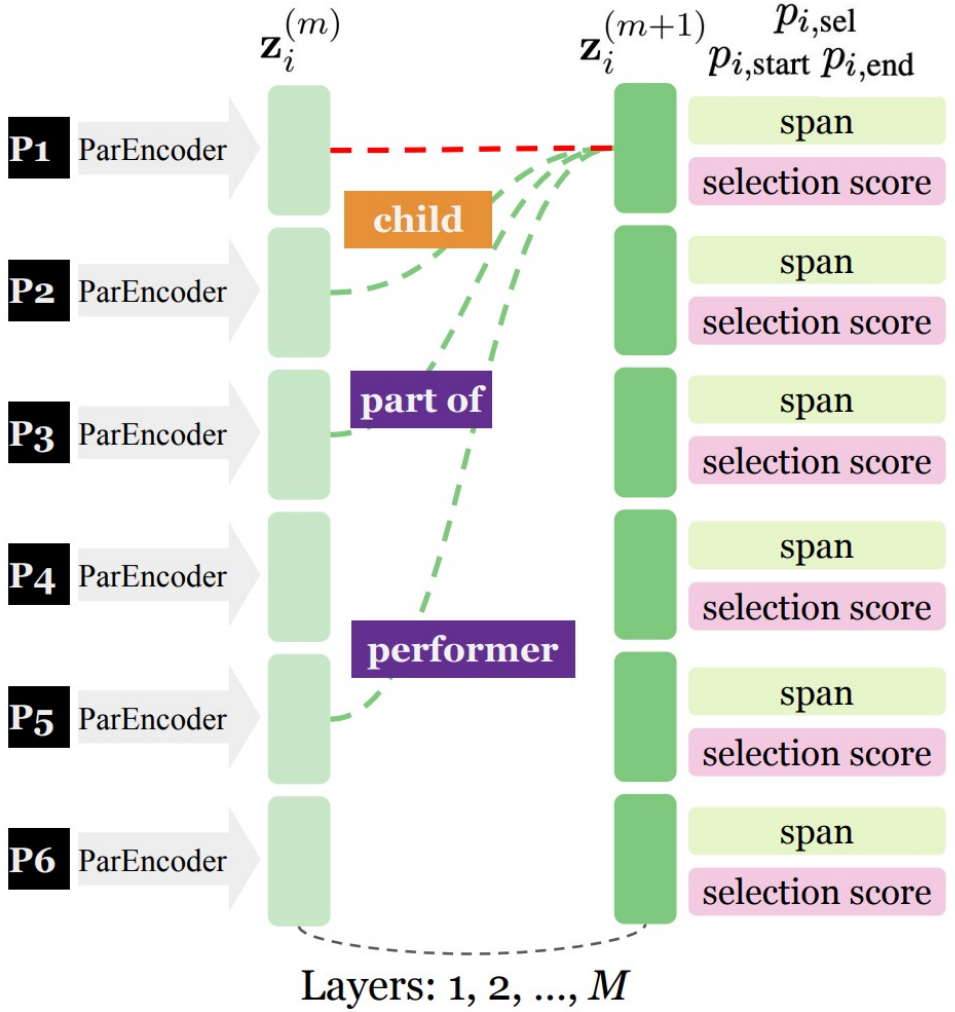
2) 해당 논문에서는 GCN을 사용함

3) Span extraction 진행

$$\sum_{i=1}^{n} \mathbb{I}[|\mathcal{S}_i| > 0] \log P_{\text{sel}}(i) + \sum_{i=1}^{n} \log \left(\sum_{(s_i, e_i) \in \mathcal{S}_i} (P_{\text{start},i}(s_i) \times P_{\text{end},i}(e_i)) \right)$$

$\mathcal{S}_i$ is a set of spans which correspond to the answer text in p_i .

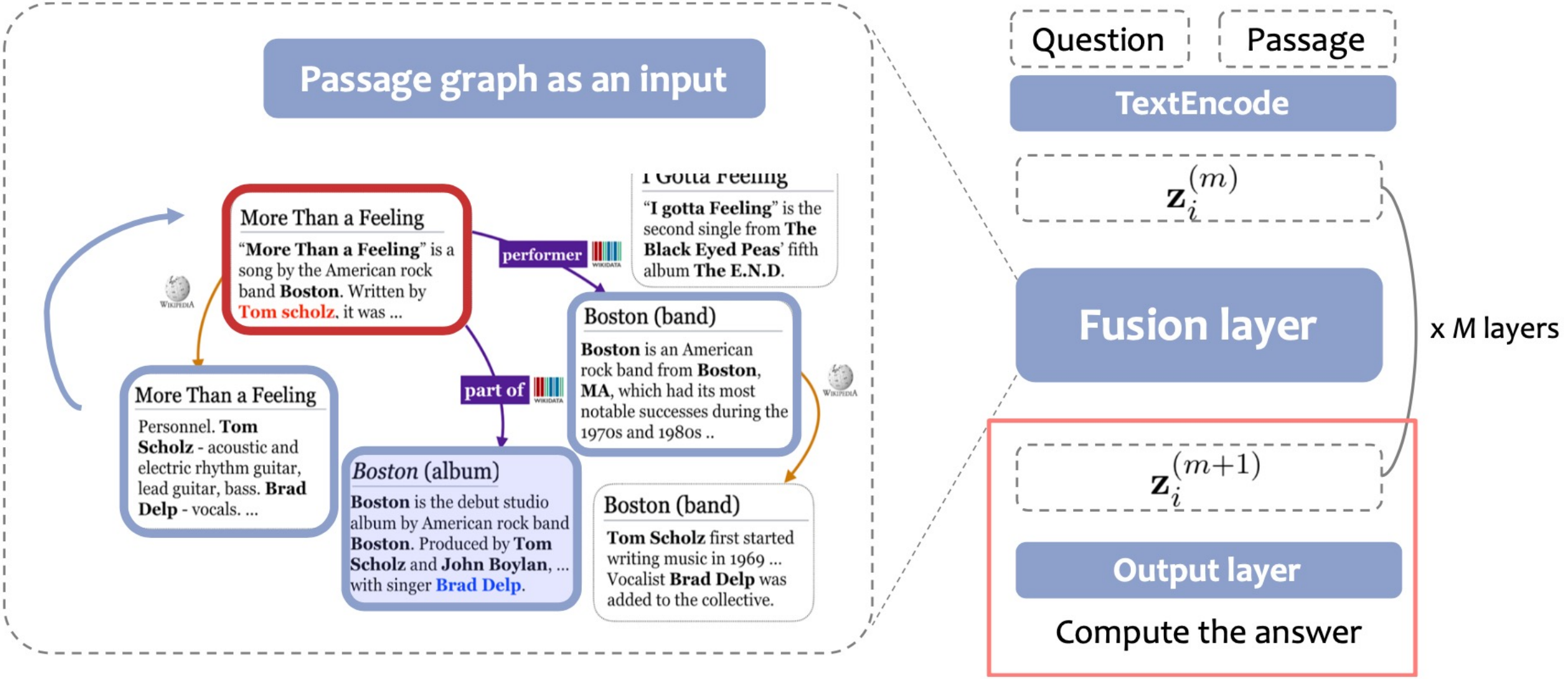
Graph Retriever



Graph Reader

03 Graph Retriever

Method



03 Graph Retriever

Result

Retriever	Reader	WebQuestions		Natural Questions		TriviaQA	
		Dev	Test	Dev	Test	Dev	Test
Text-match	ParReader	23.6	25.2	26.1	25.8	52.1	52.1
Text-match	ParReader++	19.9	20.8	28.9	28.7	54.5	54.0
GraphRetriever	ParReader	33.2	33.0	30.2	29.3	54.8	54.7
GraphRetriever	ParReader++	33.7	31.8	33.1	33.5	55.5	55.0
GraphRetriever	GraphReader (binary)	34.0	**36.4**	34.2	34.1	55.2	54.2
GraphRetriever	GraphReader (relation)	34.0	36.0	**34.7**	**34.5**	**55.8**	**56.0**
Previous best (pipeline)		-	18.5[a]	31.7[b]	32.6[b]	50.7[c]	50.9[c]
Previous best (end-to-end)		**38.5**[d]	**36.4**[d]	31.3[d]	33.3[d]	45.1[d]	45.0[d]

감사합니다