SERADE : 섹션 표현 기반 문서 임베딩 모델을 활용한 긴 문서 검색 성능 개선

정혜인○, 전현규, 김지윤, 이찬형, 김봉수
와이즈넛
{hijung, eddie14, jiyoonkim, wisnt65, usgnob}@wisenut.co.kr

SERADE: Section Representation Aggregation Retrieval for Long Document Ranking

Hye-In Jung○, Hyun-Kyu Jeon, Ji-Yoon Kim, Chan-Hyeong Lee, Bong-Su Kim
Wisenut Inc.

요 약

최근 Document Retrieval을 비롯한 대부분의 자연어처리 분야에서는 BERT와 같이 self-attention을 기반으로 한 사전훈련 모델을 활용하여 SOTA(state-of-the-art)를 이루고 있다. 그러나 self-attention 메커니즘은 입력 텍스트 길이의 제곱에 비례하여 계산 복잡도가 증가하기 때문에, 해당 모델들은 선천적으로 입력 텍스트의 길이가 제한되는 한계점을 지닌다. Document Retrieval 분야에서는, 문서를 특정 토큰 길이 단위의 문단으로 나누어 각 문단의 유사 점수 또는 표현 벡터를 추출한 후 집계함으로서 길이 제한 문제를 해결하는 방법론이 하나의 주류를 이루고 있다. 그러나 논문, 특허와 같이 섹션 형식(초록, 결론 등)을 갖는 문서의 경우, 섹션 유형에 따라 고유한 정보 특성을 지닌다. 따라서 문서를 단순히 특정 길이의 문단으로 나누어 학습하는 PARADE와 같은 기존 방법론은 각 섹션이 지닌 특성을 반영하지 못한다는 한계점을 지닌다. 본 논문에서는 섹션 유형에 대한 정보를 포함하는 문단 표현을 학습한 후, 트랜스포머 인코더를 사용하여 집계함으로서, 결과적으로 섹션의 특징과 상호 정보를 학습할 수 있도록 하는 SERADE 모델을 제안하고자 한다. 실험 결과, PARADE-Transformer 모델과 비교하여 평균 3.8%의 성능 향상을 기록하였다.

주제어: Document Retrieval, Section Representation, Transformer, self-attention mechanism

1. 서론

Document Retrieval은 키워드, 질의, 문서 등의 텍스트 입력과 관련이 있는 문서를 찾는 과제이며, Open Domain question answering(ODQA), Similar Document Retrieval 등의 다양한 분야에서 적용된다. 최근 Document Retrieval 분야에서는 BERT[1], ELECTRA[2]와 같은 사전훈련 언어모델을 활용한 모델들이 SOTA(State of the Art)를 달성하고 있다. 사전훈련 언어모델이 지닌 강점은 트랜스포머의 인코더를 사용하여 입력 문장의 문맥 표현을 학습할 수 있다는 것이다. 그러나 트랜스포머 기반 사전훈련 언어모델의 self- attention 메커니즘은 계산 복잡도가 입력 문장의 길이의 제곱에 비례하여 증가한다. 따라서 연산 비용을 줄이기 위해 입력 문장의 길이를 제한해야 한다는 한계점을 가진다[3].

이와 같은 트랜스포머 기반 사전훈련 언어모델의 입력 길이 제한 문제를 해결하기 위해 많은 선행 연구가 이루어져왔다[4][5] 사전훈련 언어모델은 문단 또는 각 문장의 연관도를 예측하기 위해 사용하며[6], 가장 점수가 높은 k개의 문단의 점수를 합산하여 최종 문서의 연관도 점수를 계산한다. 이러한 접근법은 많은 검색 벤치마크에서 SOTA를 달성하였다[4][5].그러나 대부분의 연구는 문단을 나누어 점수를 집계하는 Scoring Aggregation의 유형으로, 문단 간의 상호정보를 반영하지 못한다는 한계점을 가진다.

이에 대해서 Li et al(2020)[7]은 문단의 표현 벡터를 집계할 때 트랜스포머 인코더를 사용하여 문단의 상호정보가 반영된 문서의 표현 벡터를 생성하는 PARADE 모델을 제안하였다. 기존 연구와 다르게 문단의 표현 벡터에 대해 연관도 점수를 산출하는 것이 아니라, 문단 표현 벡터가 지닌 정보와 문단 간의 상호 정보를 학습할 수 있도록 트랜스포머의 self-attention 구조를 통해 학습하는 Representation Aggregator 방식을 제안하였다.

그러나 일반적인 문서가 아닌 논문과 특허 등의 문서의 경우 사전에 정의된 섹션이 존재하며, 해당 섹션에 따라 내용은 다른 특징을 지니고 있다. 따라서 논문 또는 특허를 위한 Document Retrieval 과제에서 PARADE 모

델을 적용한다면, 각 섹션이 지닌 고유한 특성을 반영하지 못하고 단순히 전체 문서를 일정한 문단의 길이로 나누어 정보 손실을 야기할 수 있다.

따라서 본 논문에서는 PARADE와 같은 방식으로 문단을 나누어서 전체 논문에 대한 표현을 학습하되, 논문의 섹션의 특징을 학습할 수 있도록 아래와 같은 절차로 문서의 표현을 학습하였다. 먼저, 문서의 섹션 안에서 문단을 구분하며, 이후 문단에 섹션의 유형에 따라 고유 Special Token을 부여하여 문단 표현이 섹션 유형의 정보를 포함하도록 하였다. 이후 각 문단의 표현을 트랜스포머 인코더를 통해 집계함으로써 섹션이 지닌 특성과 섹션 간의 유의미한 상호 정보를 학습할 수 있는 모델을 제안하고자 한다.

2. 관련 연구

2.1 사전훈련 모델을 활용한 Document Retrieval

BERT[1], ELECTRA[2]와 같은 사전학습 언어모델은 Document Retrieval 벤치마크에서 SOTA를 달성하고 있다. 사전학습 언어모델의 강점은 트랜드포머 인코더 구조를 이용해서 입력 텍스트의 문맥적인 표현을 학습할 수 있다는 점에 있다. 이는 word2Vec이나 GloVe와 같이 문맥의 표현을 학습할수 없는 모델과 대조되는 강점이다

사전 훈련된 언어 모델을 통한 문서 검색은 대표적으로 Bi-encoder[8][9][10] 와 Cross-encoder[11][12] 유형이 있다. Bi-encoder는 동일한 표현 공간에 질문과 답변의 표현을 각각 매핑하는 언어모델을 학습한 후, 두 벡터 간의 유사성을 평가하는 모델이다. Cross-encoder는 질문과 답변 문서 간의 유사도 점수를 산출하는 과정에서 두 텍스트를 연결하여 표현 공간에 매핑함으로서, 질문과 답변토큰 간의 상호작용을 학습할 수 있는 모델이다. Cross-encoder는 Bi-encoder에 비하여 높은 성능을 갖지만 계산량이 많으며, 미리 문서의 표현을 캐싱할 수 없으므로 Bi-encoder에 비하여 속도가 느리다 단점이 있다.

Humeau, S. et al(2019).[13] 는 위에서 언급한 두 가지 유형의 장점을 결합한 Poly-encoder를 제안하였다. 먼저 Bi-encoder 방식으로 문서를 각각 인코딩하여 캐싱한 후, 질문이 입력되면 미리 캐싱된 문서 표현에 대해 Cross-encoder 방식으로 attention 연산을 수행하여 문서의 최종 연관도 점수를 계산한다. Reimers, N, et al(2019)[14]는 문장 간 유사도를 비교할 때, Bi-encoder 방식을 차용해 속도를 개선하지만 Siamese BERT-Network를 활용하여 인코더를 하나로 통합하여 활용함으로써 파라미터의 수를 줄이는 시도를 하였다.

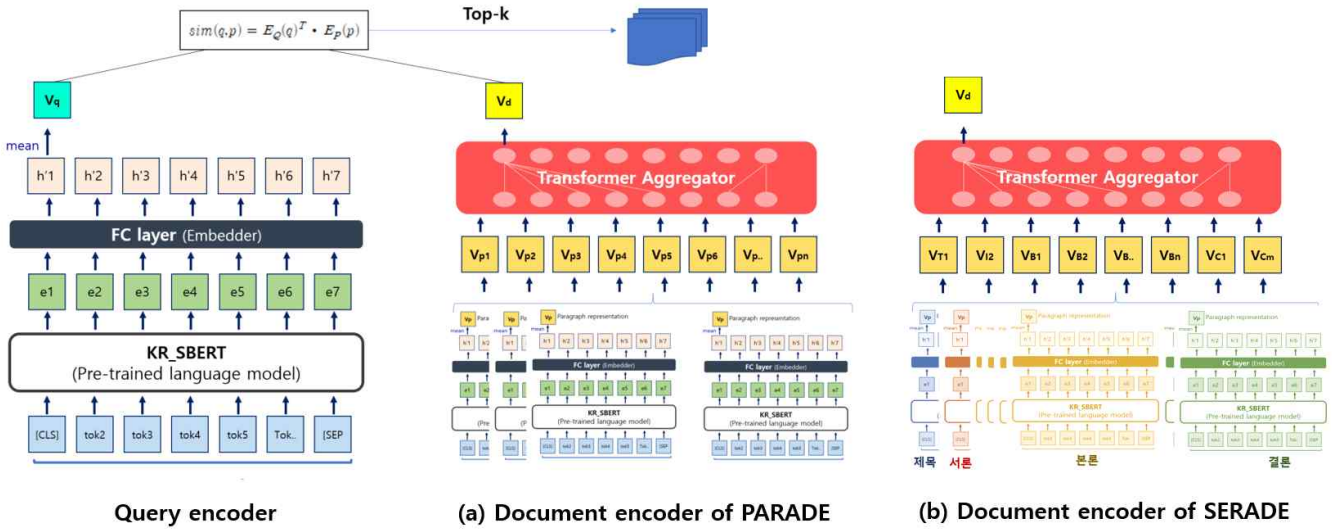
2.2 긴 문서 처리를 위한 문단 기반 모델링

BERT의 경우 최대 512 토큰의 입력 텍스트로만 훈련되었기 때문에, 512를 초과하는 토큰의 위치에 대한 position embedding은 불가능하다. 충분한 데이터가 주어진다면 파인튜닝을 하여 추가적인 position embedding을 학습할 수 있지만, 트랜스포머 기반 사전훈련 언어모델의 self- attention 메커니즘은 입력 문장의 길이의 제곱에 비례하여 계산 복잡도 증가한다. 따라서 단순히 입력가능한 토큰 길이를 늘리기 위해 더 많은 하드웨어를 할애하는 것은 현실적인 해결책이 될 수 없다[15].

이를 해결하기 위하여 문서를 문단 단위로 나누어 점수를 집계하는 문단 표현 집계 방식을 적용한 연구가 등장하였다. Akkalyoncu Yilmaz et al(2019)[16]은 문서를 문장 단위로 나누어 질문과 가장 유사한 n개 문장의 점수를 합산하는 Birch모델을 제안하였다. 또한, Dai and Callan(2019)[5]는 문서를 문단으로 나누는 과정으로 발생하는 정보 손실을 막기 위하여 중복을 허용하여 sliding 방식으로 문단을 나누는 MaxP모델을 제안하였다. 이후 MacAvaney et al(2019)[18]는 앞선 연구에서 문단 표현 벡터 자체가 가진 문맥적인 정보가 활용되지 못한다는 점에 착안하여 문단 표현 벡터를 추출함과 동시에 각 문단 표현 벡터가 질문 표현 벡터간의 연관성 정보를 합산하는 CDER 모델을 제안하였다. 더 나아가 Li et al(2020)[7]는 문단 표현과 질문 표현이 가진 상호의존 정보를 학습하는데 트랜스포머 구조를 사용하는 PARDADE 모델을 제안하여 좋은 성능을 기록하였다.

3. 연구 방법

본 연구에서는 질문과 문서의 표현을 추출하기 위하여 각각 다른 인코더 Eq, Ed를 사용하여 학습하는 Bi-encoder 방식을 적용하였다. 각 인코더를 사용하여 질문과 문서의 값을 128 차원의 벡터 공간에 매핑하며, 결과적으로 주어진 질문 벡터와 가장 가까운 k개의 문단 벡터를 검색한다. 질문과 문단의 유사도는 각 벡터들 간



[그림 1] 질문 인코더와 문서 인코더(PARADE, SERADE) 모델 구조도

의 내적으로 정의하였다.

3.1 입력층

PARADE와 같은 방식으로 문단을 나누어서 전체 논문에 대한 표현을 학습하되, 논문의 섹션의 특징을 학습할 수 있도록 섹션 내에서 문단을 구분하고자 한다. 또한, 하나의 문단 앞에 섹션 유형 별 Special Token을 추가하여 문단을 섹션에 따라 구분한다. 이를 위해, 논문 데이터에서 각 문단에 대한 섹션 정보를 확인 한후, 하나의 문서를 제목, 서론, 본론, 결론으로 구분하였다. 이후 각 섹션 내에서 문단을 나누되 제목의 경우 길이가 매우 짧다는 점을 고려하여 제목과 서론은 하나의 문단으로 분류하며 두 텍스트 사이에는 special token [CLS_I]을 넣어 구 텍스트를 구분하였다. 하나의 문서가 구성하는 문단의 수는 최대 10건으로 제한하였다.

3.2 인코더

본 연구에서는 질문과 답변 문서에 대한 인코더가 각각 존재하는 Bi-encoder 형식을 적용하였다. 두 인코더에서 사용하는 사전훈련 언어모델은 한국어에 대하여 사전훈련된 KR-SBERT[1]를 사용하였다. KR-SBERT에서는 전체 입력 토큰에 대하여 평균 풀링을 통하여 표현을 추출하며, 최종 표현 벡터의 차원은 768이다.

$p_i = KRSERT(P_i)$

Li et al(2020)[7]의 PARDADE 모델에서는 문단 인코더를 사용하여 정해진 길이로 나누어진 문단에 대한 표현을 추출하였으며 문단을 나누는 과정에서 발생하는 문맥 정보 손실을 방지하기 위하여 문단 간의 텍스트의 중복을 허용하는 stirde를 설정하였다. 본 논문에서 구현한 PARADE 모델은 전체 문서를 정해진 225 토큰 길이에 따라 window sliding을 하되 25 토큰의 stride를 설정하였다. 제안하는 SERADE 모델에서는 문단이 하나의 섹션만 가질 수 있도록 섹션 내에서 문단을 나눈 후 모든 섹션의 문단 표현을 추출하였다. 전체 문서 D는 n개의 문단 표현으로 표현되며 ($D = P_1, P_2 ..., P_n$), 하나의 문단은 인코더를 평균 풀링을 사용하여 768차원의 128차원의 벡터로 추출된다.

3.3 문단 표현의 집계

문단 표현 벡터가 지닌 정보와 문단 간의 상호 정보를 학습하기 위하여, Li et al(2020)[7]이 제안한 PARADE의 모델의 문단 표현 집계 방식을 따라 트랜스포머의 self-attention 구조를 사용하였다. 제안 모델에 사용된 트랜스포머는 head 개수는 4개, encoder의 개수는 2개로 구성되며 이후 추가 실험을 위해 encoder가 1인 모델을 학습하여 성능을 비교하였다. 트랜스포머의 입력 값은 모든 문서의 표현 벡터를 연결한 벡터(x')이며, 트랜스포머 층을 거치며 문단 간의 순서와 상호 정보를 학습한다.

1) https://github.com/snunlp/KR-SBERT

LayerNorm에서는 layer단위의 정규화[18]를 진행하며, MultiHead는 multi-head self attention을 의미한다[19]. FFN은 2개 층의 활성화함수 ReLu를 갖는 feed-forward 네트워크이다. 트랜스포머의 마지막 층의 [CLS] 토큰의 벡터가 문서를 대표하는 문서 벡터로 한다.

$$x^l = LayerNorm(x^l + MultiHead(x^l))$$
$$x^{l+1} = LayerNorm(h + FFN(h))$$

3.4 In-batch negative 학습

본 논문에서는 Vladmir K et al(2020)[20]이 제안한 in-batch negative 학습을 적용하였다. in-batch negative 학습이란 각각의 인코더 학습해서, 내적 값이 좋은 ranking 함수가 되도록 학습하는 메트릭 러닝[21] 방법론의 일종이다. In-batch negative 학습에서는 한 배치 내에 질문과 문서의 쌍이 각 문서 별 1개씩 존재하여 positive 샘플이 되며 나머지 관련이 없는 문서보다 가까운 거리를 갖 도록 인코더 학습한다. 이를 통해 각 인코더를 통해 질문과 문서의 표현이 가장 좋은 vector space를 형성하는 것이 학습의 목표이다.

$$sim(q,p) = E_Q(q)^T \bullet E_P(p)$$

임베딩의 유사도는 두 임베딩의 내적값으로 하며, 손실 함수는 유사도에 대한 Negative Log Likelihood Loss (NLL loss)를 사용하였다.

$$L(q_i, p_i^+, p_{i,1}^-, \cdots, p_{i,n}^-) = -\log \frac{e^{sim(q_i, p_i^+)}}{e^{sim(q_i, p_i^+)} + \sum_{j=1}^{n} e^{sim(q_i, p_{i,j}^-)}}$$

3.5 추론 및 평가

추론 단계에서는 문서 인코더인코더(Ep)를 통해 얻은 임베딩을 FAISS[22] 을 이용하여 index하 였다. FAISS는 매우 효율적인 오픈소스 검색 라이브러리로 유사도 검색과 밀집 벡터 클러스터링 과제에 사용되며, 수십 억 개의 벡터에 적용될 수 있다. 주어진 질문에서 vq = Eq(q) 임베딩을 추출한 후에 vq와 가장 유사한 k개의 passage를 선정하여 top-k 정확도를 산출한다.

4. 실험 및 결과

한국어 사전학습 언어모델은 KR-SBERT을 이용하였으며, 모델은 in-batch negative 방법으로 학습하여 배치 사이즈를 8로 하여 최대 20 epoch까지 학습하였다. 또한, 문단 임베딩 단계에서 과적합을 방지하기 위해 0.3 비율의 드롭아웃을 추가하였다. 최적화 알고리즘은 RAdam을 사용하여 1e-5의 학습률을 적용하였다.. 추가적으로 기울기 소실 및 폭주 문제를 방지하기 위하여 임계값은 5.0의 Gradient clipping을 적용하였다.

4.1 실험 데이터

본 연구에서는 한국과학기술정보연구원에서 제공하는 국내 논문 QA 데이터셋[23]을 활용하였다. 해당 데이터는 기계가 과학기술 문헌을 읽고 이해하는 능력을 평가하기 위해 구축된 질의응답 데이터셋으로 국내 한글 논문에 대한 질문과 정답 쌍으로 이루어져있다. 총 276,804 건의 데이터셋 중에서 논문의 4가지 섹션(제목, 서론, 본론, 결론)이 명시된 논문을 추려 총 8만 건의 학습 데이터와 1,000건의 테스트 데이터로 구성하였다.

4.2 실험 결과

표 1은 테스트 데이터셋에서 기저 모델인 PARADE-Transformer 모델과 serade 모델의 검색 성능을 비교한 결과이다. PARADE-Transformer 모델은 Top-5, Top-20, Top-50 정확도에서 평균 33.4% 성능을 기록하였으며, serade 모델은 45.9% 을 기록하여. serade 모델이 PARADE-Transformer 모델 보다 평균 12.5%의 성능 향상이 있음을 확인하였다. 또한, 두 모델의 성능의 차이는 k가 클수록 증가하는 양상을 보인다.

표 1. 테스트 데이터셋의 Top-k 검색 정확도

Type	Top-2	Top-5	Top-20
KR_SBERT	7.6	13.4	31.5
PARADE-Transformer	14.0	25.1	48.8
SERADE-Transformer	**17.6**	**29.4**	**52.2**

표 2는 serade 모델에서 섹션 문단의 표현을 집계하는 트랜스포머 내의 인코더 수에 따른 성능을 비교한 결과이다. 트랜스포머 내 1개의 인코더로 학습한 모델은 평균 20.4%의 성능을 기록하였으며, 4개의 인코더로 학습한 모델은 평균 45.9%의 성능을 기록하였다. 특히 1개의 인코더로 학습한 모델은 실험의 기저 모델인 PARADE-Transformer 모델 보다 더 낮은 성능을 보이고 있다. 이는 충분한 인코더의 수가 확보되지 않는다면 섹션 필드에 따른 구분이 검색 성능을 저하시킬 수 있음을 보여준다.

표 2. serade 모델의 Transformer 내 Encoder 수에 따른 검색 정확도

Encoder 수	Top-2	Top-5	Top-20
1	4.9	10.8	26.1
2	7.6	13.4	31.5
4	17.6	29.4	52.2

5. 결론 및 향후 연구 과제

5.1 결론

본 논문에서는 Document Retrieval에서 논문의 섹션이 지닌 특성을 반영하지 못하는 PARADE 모델 한계점을 개선하기 위하여 섹션을 기준으로 문단 표현을 집계하는 방법론을 제안하였다. 실험 결과, PARADE-Transformer 모델과 비교하여 평균 12.5%의 성능 향상을 기록하였다.

SERADE 모델 내에서 섹션 문단의 표현을 집계하는 트랜스포머 내의 인코더 수에 따른 성능을 비교한 결과, 인코더 수가 증가할수록 향상하는 것을 확인할 수 있었다. 그 중 1, 2개의 인코더로 학습한 SERADE 모델은 실험의 기저 모델인 PARADE- Transformer 모델 보다 더 낮은 성능을 보였는데, 이를 통해 충분한 인코더의 수가 확보되지 않는다면 섹션 필드에 따른 구분이 검색 성능을 저하시킬 수 있음을 확인하였다.

5.2 향후 연구 과제

본 연구는 self-attention 기반 사전훈련 모델이 지닌 고질적인 입력 텍스트 길이 제한 문제를 해결하되, 문서의 섹션이 지닌 특성과 섹션 간의 유의미한 상호 정보를 학습할 수 있는 모델을 제안한 점에서 새로운 시사점을 제공하지만, 몇 가지 연구의 한계점을 갖고 있다.

첫째, 긴 문서 처리 문제를 해결하기 위한 다양한 방법론 중 문단 단위 집계 유형의 방법론에 대해서만 다루고 있다는 한계점을 지닌다. 최근 트랜스포머 아키텍처를 수정하여 긴 텍스트를 효율적으로 처리하는 다양한 방법론이 등장하고 있다. 이와 같은 트랜스포머 아키텍처를 개선한 모델과 비교하여 SERADE 모델이 우수한 성능을 보이는지에 대한 검증이 추가적으로 이루어질 필요가 있다.

둘째, SERADE 모델의 섹션 유형 고유의 special token을 추가하는 학습 방법이 결과에 유의미한 영향을 미치는가에 대한 검증이 부족하다. 향후 연구에서는 섹션에 따라 문단을 나누되 고유한 special token과 공통된 special token을 부여한 모델을 설계한 후, 비교하여 섹션 고유 special token이 성능에 미치는 영향을 분석하고자 한다.

감사의 글

이 논문은 2019년도 정부(행정안전부)의 재원으로 국립재난안전연구원의 지원을 받아 수행된 연구임 (No.1315001260,생활안전 예방서비스를 위한 지능형 플랫폼 기술개발)

참고문헌

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In North American Association for Computational Linguistics (NAACL). 2019.

[2] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In ICLR. OpenReview.net. 2020.

[3] M. Bendersky, H. Zhuang, J. Ma, S. Han, K. Hall, and R. McDonald. RRF102: Meeting the TREC-COVID challenge with a 100+ runs ensemble. arXiv:2010.00200, 2020.

[4] Zeynep Akkalyoncu Yilmaz, Shengjin Wang, Wei Yang, Haotian Zhang, and Jimmy Lin. Applying BERT to Document Retrieval with Birch. In EMNLP. 2019.

[5] Z. Dai and J. Callan. Deeper text understanding for IR with contextual neural language modeling. In Proceedings of the 42nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2019), pages 985-988, Paris, France, 2019b.

[6] Zhuyun Dai and Jamie Callan. Deeper Text Understanding for IR with Contextual Neural Language Modeling. In SIGIR. ACM, 985-988. 2019.

[7] C. Li, A. Yates, S. MacAvaney, B. He, and Y. Sun. PARADE: Passage representation aggregation for document reranking. arXiv:2008.09093, 2020a.

[8] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. Journal of the American society for information science, 41(6):391-407, 1990.

[9] Bing Bai, Jason Weston, David Grangier, Ronan Collobert, Kunihiko Sadamasa, Yanjun Qi, Olivier Chapelle, and Kilian Weinberger. Supervised semantic indexing. In Proceedings of the 18th ACM conference on Information and knowledge management, pp. 187-196. ACM, 2009.

[10] Ledell Yu Wu, Adam Fisch, Sumit Chopra, Keith Adams, Antoine Bordes, and Jason Weston. Starspace: Embed all the things! In Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[11] Yu Ping Wu, Wei Chung Wu, Chen Xing, Ming Zhou, and Zhoujun Li. Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots. In ACL, 2017.

[12] Zhuosheng Zhang, Jiangtong Li, Pengfei Zhu, Hai Zhao, and Gongshen Liu. Modeling multi-turn conversation with deep utterance aggregation. In COLING, 2018b.

[13] Humeau, S., Shuster, K., Lachaux, M. A., & Weston, J. Poly-encoders: Transformer architectures and pretraining strategies for fast and accurate multi-sentence scoring. arXiv preprint arXiv:1905.01969. 2019.

[14] Reimers, N., & Gurevych, I. Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084. 2019.

[15] Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. 2020. Pretrained transformers for text ranking: Bert and beyond. arXiv preprint arXiv:2010.06467. 2020.

[16] Z. Akkalyoncu Yilmaz, W. Yang, H. Zhang, and J. Lin. Cross-domain modeling of sentence-level evidence for document retrieval. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 3490-3496, Hong Kong, China, 2019b.

[17] S. MacAvaney, A. Yates, A. Cohan, and N. Goharian. CEDR: Contextualized embeddings for document ranking. In Proceedings of the 42nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2019), pages 1101-1104, Paris, France, 2019a

[18] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization. CoRR abs/1607.06450. 2016.

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In NIPS. 5998-6008. 2017.

[20] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In Empirical Methods in Natural Language Processing (EMNLP), pages 6769-6781. 2020.

[21] Brian Kulis. Metric learning: A survey. Foundations and Trends in Machine Learning, 5(4):287-364. 2013.

[22] Jeff Johnson, Matthijs Douze, and Herve Jegou. Billion-scale similarity search with GPUs. ArXiv. abs/1702.08734. 2017.

[23] 한국과학기술정보연구원 : 국내 논문 QA 데이터셋. Version 1.1. 한국과학기술정보연구원. https://doi.org/10.23057/49. 2021.