사물인터넷융복합논문지 Vol. 8, No. 5, pp. 17-25, 2022 https://doi.org/10.20465/KIOTS.2022.8.5.017

HTML 태그 깊이 임베딩: 웹 문서 기계 독해 성능 개선을 위한 BERT 모델의 입력 임베딩 기법

목진왕[1], 장현재[2], 이현섭[3*]
[1]백석대학교 컴퓨터공학부 학부생, [2]백석대학교 컴퓨터공학부 교수, [3]백석대학교 첨단IT학부 교수

HTML Tag Depth Embedding: An Input Embedding Method of the BERT Model for Improving Web Document Reading Comprehension Performance

Jin-Wang Mok[1], Hyun Jae Jang[2], Hyun-Seob Lee[3*]
[1]Student, Division of Computer Engineering , Baekseok University
[2]Professor, Division of Computer Engineering , Baekseok University
[3]Professor, Division of Advanced IT, Baekseok University

요 약 최근 종단 장치(Edge Device)의 수가 증가함에 따라 빅데이터가 생성되었고 특히 정제되지 않은 HTML 문서가 증가하고 있다. 따라서 자연어 처리 모델을 이용해 HTML 문서 내에서 중요한 정보를 찾아내는 기계 독해(Machine Reading Comprehension) 기술이 중요해지고 있다. 본 논문에서는 기계 독해의 여러 연구에서 준수한 성능을 보이는 BERT(Bidirectional Encoder Representations from Transformers) 모델이 HTML 문서 구조의 깊이를 효과적으로 학습할 수 있는 HTDE(HTML Tag Depth Embedding Method)를 제안하였다. HTDE는 BERT의 각 입력 토큰에 대하여 HTML 문서로부터 태그 스택을 생성하고 깊이 정보를 추출한다. 그리고 BERT의 입력 임베딩에 토큰의 깊이를 입력으로하는 HTML 임베딩을 더한다. 이 방법은 문서 구조를 토큰 단위로 표현하여 주변 토큰과의 관계를 식별할 수 있기 때문에 HTML 문서에 대한 BERT의 정확도를 향상시키는 효과가 있다. 마지막으로 실험을 통해 BERT의 기존 임베딩 기법에 비해 HTML 구조에 대한 모델 예측 정확도가 향상됨을 증명하였다.

주제어 : 자연어 처리, 기계 독해, 임베딩, HTML, BERT

Abstract Recently the massive amount of data has been generated because of the number of edge devices increases. And especially, the number of raw unstructured HTML documents has been increased. Therefore, MRC(Machine Reading Comprehension) in which a natural language processing model finds the important information within an HTML document is becoming more important. In this paper, we propose HTDE(HTML Tag Depth Embedding Method), which allows the BERT to train the depth of the HTML document structure. HTDE makes a tag stack from the HTML document for each input token in the BERT and then extracts the depth information. After that, we add a HTML embedding layer that takes the depth of the token as input to the step of input embedding of BERT. Since tokenization using HTDE identifies the HTML document structures through the relationship of surrounding tokens, HTDE improves the accuracy of BERT for HTML documents. Finally, we demonstrated that the proposed idea showing the higher accuracy compared than the accuracy using the conventional embedding of BERT.

Key Words : Natural Language Processing, Machine Reading Comprehension, Embeddings, HTML, BERT

*본 논문은 2022년도 교육부의 재원으로 한국연구재단의 지원을 받아 수행된 지자체-대학 협력기반 지역혁신 사업(2021RIS-004), 기초연구 사업(NRF-2021R1I1A3061020)과 과학기술정보통신부의 재원으로 한국 연구재단의 지원(NRF-2021R1C1C2012843)을 받아 수행되었음.
*교신저자 : 이현섭(hyunseob@bu.ac.kr)
접수일 2022년 7월 18일 수정일 2022년 9월 2일 심사완료일 2022년 9월 6일

1. 서론

최근 사물인터넷 기기, 스마트폰과 같은 종단 장치(Edge Device)의 수가 증가하고 있고 관련 연구가 진행되고 있다.[1, 2, 3] 이와 같은 종단 장치들은 빅데이터라고 부르는 방대한 양의 데이터를 생성하는데 그중에서도 HTML과 같이 구조화된 문서 데이터가 급격하게 증가하고 있다. 이러한 문서 데이터는 텍스트(Text) 뿐만 아니라 태그(Tag)와 속성(Attribute) 정보를 포함하고 있기 때문에 기계가 자동으로 식별하고 이해하기 위해 자연어 처리의 한 분야인 기계 독해의 중요도가 상승하고 있다. 기계 독해는 기계 학습 모델이 주어진 지문을 바탕으로 질의 응답(Question Answering)을 수행하는 연구 분야로 최근까지 연구가 활발히 진행되고 있다.[4, 5, 6, 7]

질의 응답에 대한 연구에서 높은 성능을 보여주는 대표적인 모델로 BERT[8]가 있다. BERT는 양방향 사전학습 모델로 여러 태스크에서 범용적으로 좋은 성능을 보여주기 때문에 질의 응답 성능을 향상시키기 위한 연구 분야에서 다양하게 활용되고 있다. BERT의 위치 임베딩 연구[9]에 따르면 BERT의 위치 임베딩 기법은 절대 위치 임베딩 기법(Absolute Position Embedding Method)과 상대 위치 임베딩 기법(Relative Position Embedding Method)이 있으며 상대 위치 임베딩 기법을 사용할 때 질의 응답에서 더 좋은 성능을 내는 것이 입증되었다.

질의 응답 데이터 셋은 대표적으로 영문 질의 응답 데이터 셋인 SQuAD(StanfordQuestion Answering Dataset)[10] 와 HotPotQA(Hot Pot Question Answering Dataset)[11] 등이 있으며 한국어 질의 응답 데이터 셋으로는 LG CNS에서 개발한 KorQuAD(Korean Question Answering Dataset) 1.0[12]과 KorQuAD 2.0[13]이 있다. KorQuAD 1.0은 약 한 문단 길이의 짧은 평문 형식의 지문과 질의 응답으로 구성된 데이터 셋이다. 그리고 KorQuAD 2.0은 HTML로 표현된 위키피디아(Wikipedia) 문서를 이용하여 질의 응답을 수행하는 데이터 셋으로 질의 응답에서 빈번하게 사용되는 웹 문서 구조에 대해 자연어 처리 모델이 학습할 수 있는 데이터 셋이다.

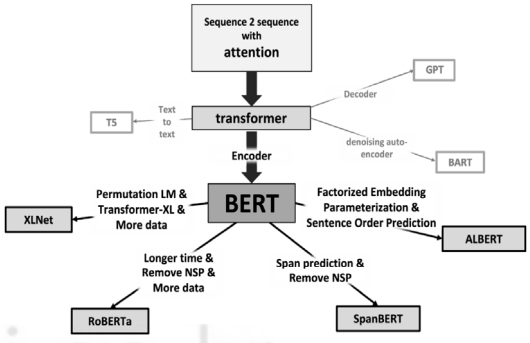
LG CNS는 KorQuAD 2.0 데이터 셋에 대한 실험의 기준 모델로 절대 위치 임베딩이 적용된 BERT multilingual-base-cased(이하 BERT) 모델을 사용했다. 그러나 기준 모델은 절대 위치 임베딩 기법만으로 학습을 진행하므로 HTML 문서 구조를 파악하기 어렵다는 한계가 있다. 따라서 본 논문에서는 효과적인 HTML 문서 구조 학습을 위해 절대 위치 임베딩에 HTML 태그 깊이 정보를 더한 HTML 태그 깊이 임베딩 기법(HTDE)을 제안한다. 본 논문에서 제안하는 HTDE는 입력 시퀀스의 각 토큰에 대한 HTML내의 깊이를 구해 입력 임베딩에 결합하여 절대 위치 임베딩만 사용한 BERT 모델 뿐만 아니라 상대 위치 임베딩을 사용한 BERT 모델보다도 좋은 성능을 보여준다.

실험은 KorQuAD 2.0 데이터 셋으로 진행되며 기준 모델인 절대 임베딩을 적용한 모델 그리고 질의 응답 문제에서 좋은 성능을 보이는 상대 임베딩을 적용한 모델과 성능을 비교하여 제안한 기법의 효과를 증명한다.

2. 관련연구

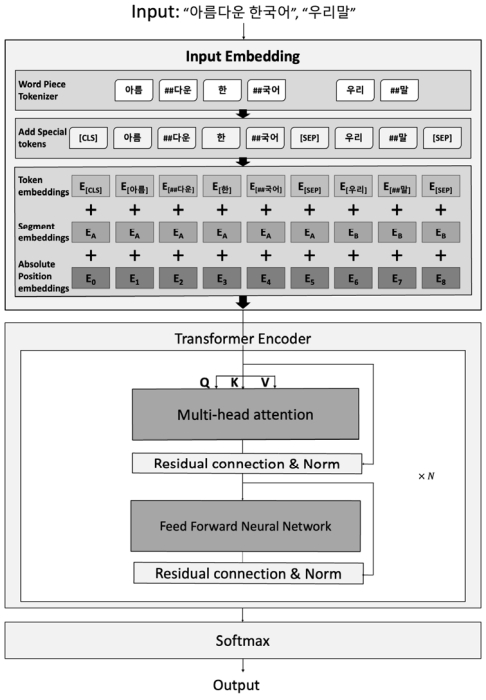
2.1 Bidirectional Encoder Representation from Transformers(BERT)

과거의 기계 번역 모델들은 RNN(Recurrent Neural Network)을 기반으로 한 인코더-디코더 구조를 가지고 있었다[14]. 이 모델들은 문장이 길어질수록 문장의 앞에 위치한 정보들의 손실이 발생하는 문제점을 가지고 있었고 이를 해결하기 위해 seq2seq with attention[15]에서 어텐션(Attention) 메커니즘을 제안했다. 어텐션 메커니즘은 고정된 크기의 단일 벡터로 모든 문맥 정보를 표현하는 것이 아니라 입력 문장의 각 토큰에 대해 구별된 문맥 벡터로 표현하는 기법으로 문장 길이의 제한에서 자유롭고 각 토큰 간의 상관 관계를 효과적으로 학습할 수 있다는 장점이 있다. 따라서 최근 어텐션 메커니즘을 적용한 트랜스포머[16] 구조를 기반으로 하는 자연어 처리 모델의 연구가 활발히 진행되고 있다.



[Fig. 1] The hisory of NLP model based on attention

Fig.1은 어텐션 메커니즘을 기반으로 하는 자연어 처리 모델이 발전되어 온 과정을 보여주고 있다. 트랜스포머는 기존 RNN 기반 모델들의 순차적인 구조를 사용하지 않고 어텐션 메커니즘만을 활용한 인코더-디코더 구조를 재구성함으로써 계산 효율성을 높이고 모델의 성능을 향상시켰다. 따라서 최근 좋은 성능을 보여주고 있는 GPT3(GenerativePre-trained Trasformer), T5(Text to Text Transfer Transformer), BERT, XLNet, BART등 다양한 자연어 처리 모델들이 트랜스포머 구조를 기반으로 연구되고 있다.



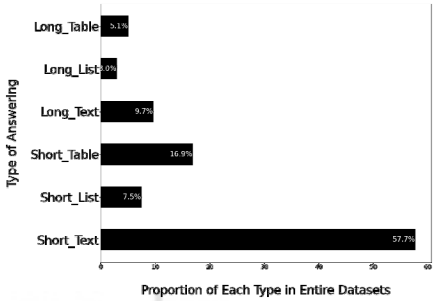
[Fig. 2] The architecture of BERT

Fig.2는 정보 검색 등 다양한 분야에서 활용되고 있는 BERT의 구조를 보여주고 있다. BERT는 크게 입력 임베딩 모듈, 트랜스포머 기반 인코더 모듈, 소프트맥스(Softmax) 모듈로 구성되어 있다. 입력 임베딩 모듈에서는 입력 문자열을 워드 피스(Word Piece) 토크나이저로 토큰화하고 시퀀스의 맨 앞에 분류(Classification) 작업을 위한 [CLS] 토큰과 두 문장 사이를 구분짓기 위한 [SEP] 토큰 등 특수한 토큰을 추가한다. 그리고 이렇게 만들어진 시퀀스에 대해 토큰(Token), 세그먼트(Segment), 위치(Position) 임베딩을 진행하고 그 출력 값을 더해 입력 임베딩 벡터를 만든다. 트랜스포머 기반 인코더 모듈에서는 입력 임베딩 벡터를 쿼리(Query), 키(Key), 값(Value)으로 나눈 뒤 어텐션을 진행한다. 그리고 그 출력 값에 잔여 연결(Residual connection) 기법과 층 정규화(Layer Normalization) 기법을 적용해 순방향 신경망(Feed Forward Neural Network, FFNN)의 입력으로 전달한다. 순방향 신경망의 출력 값도 동일하게 잔여 연결 기법과 층 정규화 기법을 적용한 뒤 출력 레이어인 소프트맥스 모듈로 전달한다. 마지막으로 소프트맥스 모듈에서 전달받은 입력의 소프트맥스 값을 최종적으로 출력한다.

BERT는 전이 학습(Transfer learning) 기법을 통해 학습을 진행한다. 전이 학습은 모델을 범용적으로 학습시키는 사전학습(Pre-training) 과정과 구체적인 태스크로 모델을 조정하는 미세조정(Fine-tuning) 과정으로 진행된다. BERT의 사전학습 과정은 Masked Language Modeling(MLM)과 Next Sentence Prediction(NSP) 기법을 사용한다. MLM은 입력 시퀀스의 토큰 중 15%를 랜덤하게 [MASK] 토큰으로 변경하고 이를 맞추도록 학습시키는 기법이다. NSP는 두 문장으로 이루어진 입력에 대해 뒷 문장이 앞의 문장 다음에 나오는 것인지 참과 거짓으로 분류하는 기법으로 이를 위해 뒷 문장 중 50%는 말뭉치(Corpus) 중 앞 문장 뒤에 나오는 문장으로 배치하고 나머지 50%는 말뭉치 중 임의로 선택한 문장을 배치한다. 미세조정 과정은 기계 번역, 질의 응답 등 각각의 목적의 맞도록 입력과 출력 레이어를 조정하여 학습을 진행한다. 본 논문은 사전학습된 BERT 모델을 한국어 질의 응답 데이터 셋인 KorQuAD 2.0에 대해 미세조정하는 기법 중 하나인 HTDE를 제안하고 있다.

2.2 Korean Question Answering Dataset

한국어의 자연어 처리 모델 성능 평가를 위한 대표적인 벤치마크 데이터 셋으로 KorQuAD가 있다.



[Fig. 3] The data type of answering in KorQuad 2.0

KorQuAD 2.0 은 2020년 LG CNS에서 공개한 한국어 질의 응답 데이터 셋으로 앞서 공개한 KorQuAD 1.0 데이터 셋과 달리 지문(Context)이 일반 평문 형식이 아닌 HTML 문서 형식이라는 차이가 있다.

Fig.3은 KorQuAD 2.0의 응답 유형 비율을 보여주고 있다. KorQuAD 2.0의 전체 데이터 셋은 질의 기준으로 총 102,960개이며 모델 학습을 위한 Train 데이터 83,486개와 모델 검증을 위한 Dev 데이터 10,165개 그리고 모델의 평가를 위한 Test 데이터 9,309개로 이루어져 있다. 또한 응답의 경우 짧은 평문(Short Text) 유형이 전체 응답 데이터의 57.7%로 가장 많고 이어서 짧은 리스트(Short List) 유형 7.5%, 짧은 표(ShortTable) 유형 16.9%, 긴 평문(Long Text) 유형 9.7%, 긴 리스트(Long List) 유형 3.0%, 긴 표(LongTable) 유형 5.1%로 구성되어 있다.

KorQuAD 2.0의 모델 평가 지표는 대표적인 영문 질의 응답 데이터 셋인 SQuAD, HotPotQA, TriviaQA 등에서 활용되었던 EM(Exact Match)과 F1 점수를 사용한다. EM은 모델의 예측 결과가 실제 값과 정확히 일치하는지를 0또는 1로 분류하고 분류된 데이터 중 1의 비율을 나타낸 평가 지표이다. 그리고 F1 점수는 정밀도(Precision)와 재현율(Recall)의 조화 평균으로 모델의 예측 결과와 실제 값의 유사도를 나타내는 평가 지표이다.

2.3 위치 임베딩(Position Embedding)

재귀적인 구조로 자연스럽게 순서 정보를 학습하는 RNN 기반 모델들과는 다르게 어텐션 매커니즘을 활용한 모델들은 위치 임베딩 기법을 통해 순서 정보를 학습한다. 특히 BERT를 활용한 자연어 처리 연구들이 활발하게 진행되면서 BERT의 위치 임베딩 기법에 대한 다양한 연구들이 진행되었다. BERT의 위치 임베딩 연구에서는 위치 임베딩 기법을 절대 위치 임베딩 기법(Absolute Position Embedding)과 상대 위치 임베딩 기법(Relative Position Embedding)으로 구분한다. 절대 위치 임베딩 기법은 트랜스포머와 BERT에서 사용한 기법으로 주기함수 또는 순차적인 정수를 인코딩하여 시퀀스에 절대적인 순서 정보를 부여하는 임베딩 기법이고 상대 위치 임베딩 기법은 순서에 관한 정보 없이 오프셋(Offset)을 설정해 단어 간의 상대적인 위치를 학습하는 임베딩 기법이다.

위치 임베딩의 주요한 속성은 단조성(Monotonicity), 번역 불변성(Translation invariance), 대칭성(Symmetry) 성이 있다.

$$\forall\, x,m,n \in \mathbb{N} : m > n \Leftrightarrow \phi(\overrightarrow{x}, \overrightarrow{x+m}) < \phi(\overrightarrow{x}, \overrightarrow{x+n})$$

Eq. 1 Monotonicity

$$\forall\, x_1,\ldots,x_n \in \mathbb{N} : \phi(\overrightarrow{x_1}, \overrightarrow{x_1+m}) = \phi(\overrightarrow{x_2}, \overrightarrow{x_2+m}) = \ldots = \phi(\overrightarrow{x_n}, \overrightarrow{x_n+m})$$

Eq. 2 Translation Invariance

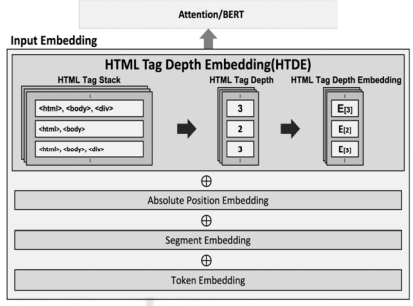
$$\forall\, x,y \in \mathbb{N} : \phi(\overrightarrow{x}, \overrightarrow{y}) = \phi(\overrightarrow{y}, \overrightarrow{x})$$

Eq. 3 Symmetry

$\overrightarrow{x} \in \mathbb{R}^D$가 $x \in \mathbb{N}$의 D차원 유클리디안 공간에서의 표현이고 함수 $\phi(\cdot\ ,\cdot\ )$가 $\overrightarrow{x}, \overrightarrow{y} \in \mathbb{R}^D$에 대한 근접성(Proximity)을 계산할 때 Eq.1과 같이 단조성은 입력 시퀀스 내의 임의의 두 벡터에 대하여 두 벡터의 거리가 멀수록 근접성이 감소하는 것을 의미한다. 그리고 Eq.2와 같이 번역 불변성은 입력 시퀀스 내의 동일한 거리를 갖는 모든 임의의 두 벡터가 동일한 근접성을 가지는 것을 의미한다. 마지막으로 Eq.3과 같이 대칭성은 입력 시퀀스 내의 임의의 두 토큰에 대하여 서로 위치를 바꾸어 계산해도 동일한 근접성을 가지는 것을 의미한다. 질의 응답과 같은 범위 예측(Span Prediction) 태스크에서는 위 속성 중 단조성, 번역 불변성 그리고 비대칭성(Asytmmetry)이 성능에 영향을 주므로 상대 위치 임베딩 기법이 절대 위치 임베딩 기법보다 나은 성능을 보여준다. 따라서 본 논문에서는 제안하는 HTDE 기법을 BERT의 절대 위치 임베딩 뿐만 아니라 상대 위치 임베딩과도 비교하여 HTML 문서의 질의 응답 문제에 대한 성능을 측정하고자 한다.

3. HTML 태그 깊이 임베딩 기법(HTDE)

3.1 핵심 아이디어



[Fig. 4] Structure of HTDE

Fig.4는 HTDE의 전체적인 구조를 보여주고 있다. BERT의 절대 위치 임베딩은 0부터 시퀀스의 최대 길이까지 1씩 커지는 일반적인 정수 인덱스 값만 위치정보를 나타내는 입력으로 사용하는 반면 HTDE는 정수 인덱스 값과 더불어 HTML 태그 스택으로부터 추출한 태그 깊이 값을 입력 임베딩에 추가한다. 태그 스택은 주어진 HTML 문서를 토큰화한 토큰 시퀀스에서 각 토큰의 위치에 해당하는 태그와 조상 태그를 저장한 스택을 의미한다. 따라서 어텐션 기반의 모델이 HTML 문서의 태그 구조를 효과적으로 학습할 수 있다.

$$\mathrm{INPUT}(x) = \mathrm{T}(x) + \mathrm{S}(x) + \mathrm{AP}(x) + \mathrm{HTDE}(x)$$

Eq. 4 Input Embedding

$$HTDE(x_i) = \lambda(STACK(x_i))$$

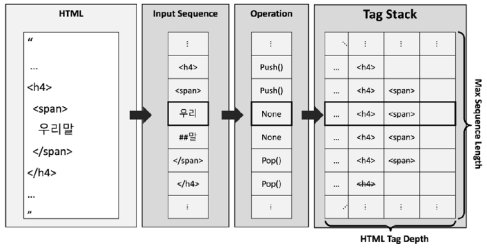
Eq. 5 HTML Tag Depth Embedding

Eq.4는 본 논문에서 제안하는 HTDE를 반영한 입력 임베딩을 구하는 식이다. 입력 토큰 시퀀스를 x라고 할 때 $T(x)$는 토큰 임베딩, $S(x)$는 세그먼트 임베딩, $AP(x)$는 절대 위치 임베딩을 의미하고 $HTDE(x)$는 HTML Tag 깊이 임베딩을 의미한다. $\lambda(x)$가 입력 시퀀스의 길이를 구하는 함수라고 할 때 Eq.5는 입력 토큰 시퀀스의 i번째 원소에 대한 HTDE 값을 구하는 식이다. $STACK(x_i)$는 입력 토큰 시퀀스 x의 i번째 원소에 대한 태그 스택을 의미한다.

3.2 HTML 태그 스택

HTML 문서는 상위 태그 내에 여러 하위 태그를 포함하는 트리 구조의 문서이며 태그 이름의 앞뒤를 '<'로 시작해 '>'로 끝나는 여는 태그와 '</' 로 시작해 '>'로 끝나는 닫는 태그로 감싼 태그쌍들 구성되어 있다. 또한 태그쌍 내에 태그쌍이 들어갈 수 있으며 이 때 바깥쪽 태그쌍을 부모 태그 안쪽 태그쌍을 자식 태그라고 한다. 이러한 관계는 트리 구조로 나타낼 수 있고 각 노드는 문서 내에서 특정한 깊이(Depth)값을 갖는다.

Fig.5는 본 논문에서 제안한 태그 스택(Tag Stack)을 보여주고 있다. 태그 스택은 HTML 문서 내 임의의 위치에서 상위 부모 태그들의 정보를 효과적으로 표현할 수 있다. 입력 시퀀스에서 태그 스택을 추출하려면 여는 태그에서 푸쉬(Push) 연산을 수행하여 태그 값을 태그 스택에 저장하고 닫는 태그에서 팝(Pop) 연산을 수행하여 가장 최근에 저장된 태그를 제거한다. 이때 〈/〉, 〈meta〉, 〈img/〉와 같이 의미가 없거나 질의 응답 문제에 도움이 되지 않는 태그들에 대해서는 연산을 진행하지 않는다. 이처럼 생성된 태그 스택의 원소의 개수는 해당 토큰의 HTML 태그 깊이 값이며 전체 시퀀스에 대한 HTML 태그 깊이 값은 절대 위치 임베딩의 입력 시퀀스의 길이와 동일하다.



[Fig. 5] Tag stack

3.3 HTDE의 장점

본 논문에서 제안하는 HTDE는 입력 임베딩에 절대 위치 임베딩만 있는 경우와 비교할 때 문장의 순서뿐 아니라 HTML 문서의 태그 구조까지 반영한다. 예를 들어, 기존 BERT 모델에서 "〈ul〉〈li〉홍길동〈/li〉〈/ul〉"이라는 문자열은 ['<', 'ul', '>', '<', 'li', '>', '홍', '##길', '##동', '<', '/', 'li', '>', '<', '/', 'ul', '>']와 같이 토큰화되고 절대 위치 임베딩 값으로 [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]와 같은 값을 갖게 된다. 그러나 HTDE는 여기에 [1, 1, 1, 2, 2, 2, 2, 2, 2, 1, 1, 1, 1, 0, 0, 0, 0]와 같은 태그 깊이 값을 추가로 갖게 되므로 모델이 〈li〉와 같은 태그 구조의 패턴을 파악할 수 있어 좋은 성능을 기대할 수 있다.

HTDE의 경우 연산량이 소폭 증가하지만 주변 토큰들이 모두 동일한 태그 내에 위치한 경우를 고려하면 주변 토큰들의 위치 정보만을 고려하는 상대 위치 임베딩보다 문서 구조를 더 정확하게 반영할 수 있으므로 HTML에 대한 질의 응답과 같은 범위 예측 태스크에서 상대 위치 임베딩보다 좋은 성능을 기대할 수 있다.

4. 실험 및 결과

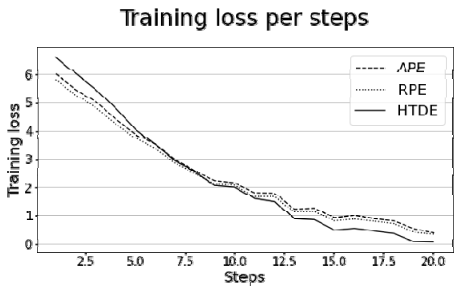
본 논문에서는 BERT 모델의 입력 임베딩으로 절대 위치 임베딩 기법, 상대 위치 임베딩 기법 그리고 본 논문에서 제안하는 HTDE 기법을 상호 비교하는 실험을 진행했다. 입력 시퀀스의 최대 길이는 512, 입력 시퀀스

의 길이가 BERT의 입력 길이를 초과할 경우를 위해 보폭(Stride)은 128, 배치 크기(Batch size)는 8, 에폭 수(Epochs)는 3, 학습률(Learning Rate)은 2e-5으로 설정했다. 그리고 HTDE가 적은 학습 데이터로도 효과가 있음을 입증하고자 전체 데이터를 128, 32로 샤딩(Sharding)한 데이터를 사용하여 데이터의 크기를 동일하게 제한했다. 성능 평가 지표로는 정답을 정확히 맞춘 비율을 의미하는 EM(Exact Match) 점수와 모델의 예측 결과와 실제값의 유사도를 의미하는 F1 점수를 이용하였다.

〈Table 1〉 Comparison between HTDE and other embedding types for EM and F1

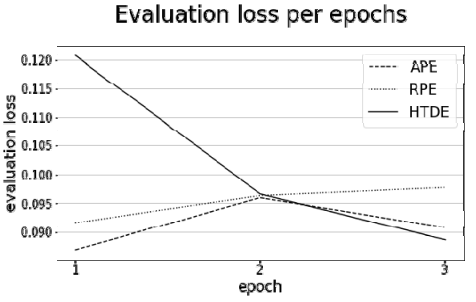
Shard	Embedding Type	EM	F1
128	APE	15.00	21.72
	RPE	21.25	27.06
	HTDE	23.75	28.68
32	APE	26.72	33.82
	RPE	30.50	37.02
	HTDE	33.64	39.33

Table 1은 BERT의 위치 임베딩에 절대 위치 임베딩과 상대 위치 임베딩 그리고 HTDE를 적용했을 때의 결과를 데이터 셋을 분할한 개수(Shard)에 따라 비교하여 보여주고 있다. 이때 샤드 수가 의미하는 것은 원본 데이터를 얼마나 작은 크기로 나눈 것인지를 의미하며 높을수록 더 적은 양의 데이터를 학습한 것을 의미한다. 절대 위치 임베딩 기법의 경우 샤드 수가 128일 때 EM 점수 약 15.00, F1 점수 21.72를 기록했고 샤드 수가 32일 때 EM 점수 약 26.72, F1 점수 약 33.82를 기록했다. 상대 위치 임베딩 기법의 경우 샤드 수가 128일 때 EM 점수 약 21.25, F1 점수 약 27.06을 기록했고 샤드 수가 32일 때 EM 점수 약 30.50, F1 점수 약 37.02을 기록했다. 본 논문에서 제안한 HTDE는 샤드 수가 128일 때 EM 점수 약 23.75, F1 점수 약 28.68을 기록했고 샤드 수가 32일 때 EM 점수 약 33.64, F1 점수가 약 39.33을 기록했다. HTDE 기법은 샤드 수가 128일 때 절대 위치 임베딩보다 EM 점수가 약 58.3%, F1 점수가 약 32% 높고 상대 위치 임베딩보다 EM 점수가 약 11.7%, F1 점수가 약 5.9% 높은 결과를 기록했다. 이는 HTDE가 다른 임베딩 기법에 비해 적은 데이터로 학습해도 HTML 문서에 대해 효과적으로 학습한다는 것을 보여준다. 또한 샤드 수가 32일 때 절대 위치 임베딩보다 EM 점수가 약 25.9%, F1 점수가 약 16.2% 높고 상대 위치 임베딩보다 EM 점수가 약 10.2%, F1 점수가 약 6.2% 높은 결과를 기록했다. 이를 통해 데이터의 양이 증가하더라도 여전히 효과적인 기법임을 입증했다.



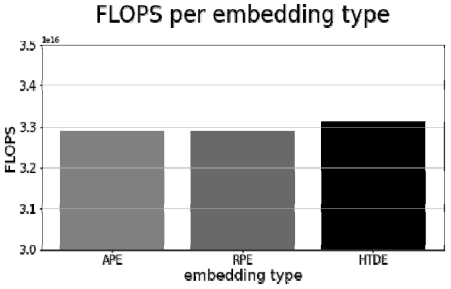
[Fig. 6] Training loss per steps

Fig.6은 전체 데이터에 대해 샤드 수 128로 20번의 스텝만큼 학습을 진행하고 각 기법의 학습 손실을 y축, 학습 스텝을 x축으로 나타낸 것이다. 절대 위치 임베딩 기법을 적용한 경우의 최초 학습 손실은 약 6.02, 최종 학습 손실은 약 0.40을 기록했고 상대 위치 임베딩 기법을 적용한 경우의 최초 학습 손실은 약 5.82, 최종 학습 손실은 약 0.35를 기록했다. 이와 달리 HTDE 기법을 적용한 경우의 최초 학습 손실은 약 6.61로 다른 두 기법에 비해 높지만 최종 학습 손실은 약 0.06으로 절대 위치 임베딩 기법보다 약 6.66e2% 수준으로 낮고, 상대 위치 임베딩 기법보다 약 5.83e2% 수준으로 낮다. 최초 학습 손실이 다른 두 임베딩 기법보다 높은 것은 HTDE 기법이 HTML 태그 깊이 임베딩 레이어를 추가하면서 학습되지 않은 파라미터들이 발생했기 때문으로 해석할 수 있다. 그러나 이후 학습이 진행되면서 HTDE 기법이 모델의 HTML 문서 이해를 돕기 때문에 빠른 속도로 손실을 줄이는 것을 확인할 수 있다.



[Fig. 7] Evaluation loss per steps

Fig.7은 Fig.6과 마찬가지로 전체 데이터에 대해 샤드 수 128로 3 에폭만큼 학습을 진행하고 각 기법의 평가 손실을 y축, 에폭을 x축으로 나타낸 것이다. 절대 위치 임베딩의 경우 평가 손실이 약 0.086에서 시작해서 점차 증가하다가 다시 감소하여 약 0.091을 기록했고 상대 위치 임베딩의 경우도 평가 손실이 약 0.091에서 시작하지만 다시 감소하지 않고 증가하여 약 0.097을 기록했다. 그러나 HTDE 기법의 경우 최초 평가 손실이 약 0.121로 다른 임베딩 기법들에 비해 높지만 최종 평가 손실은 가장 낮은 값인 약 0.088을 기록했다. HTDE 기법의 최종 평가 손실을 다른 위치 임베딩 기법들과 비교하면 절대 위치 임베딩 기법에 비해 약 2.33%, 상대 위치 임베딩 기법에 비해 약 9.26% 감소했다. 최초 평가 손실이 다른 임베딩 기법들에 비해 높은 것은 앞서 설명한 바와 같이 HTML 태그 깊이 임베딩 레이어를 추가하면서 학습되지 않은 파라미터들이 발생했기 때문으로 해석할 수 있다.



[Fig. 8] FLOPS per embedding type

Fig.8은 임베딩 기법에 따른 FLOPS(FLoating Point Operations Per Second)를 보여주고 있다. HTDE는 다른 두 임베딩 기법과 비교할 때 FLOPS가 약 0.716% 만큼 증가한다. 이러한 원인은 HTML 태그 깊이 임베딩을 추가하면서 모델의 복잡도가 올라갔기 때문이다. 상대 위치 임베딩 기법의 경우 작업량의 증가 없이 모델의 정확도를 향상시켰다. HTDE 기법의 경우 상대 위치 임베딩보다 좋은 성능을 보였으나 추가적인 HTML 태그 데이터 처리로 인해 작업량이 증가하는 경향이 발견되었다. 정확도와 작업량이 트레이드 오프 관계이지만 추가적인 실험을 통해 본 논문에서 제안하는 HTDE 기법을 적용한 모델의 정확도 증가량이 상대 위치 임베딩 기법을 적용한 모델의 경우보다 EM과 F1 점수 기준으로 각각 약 10.2~11.7%, 5.9~6.2% 증가한 것에 대비하여 작업 증가량은 약 0.716% 수준으로 소폭 증가하는 것을 확인할 수 있었다. 따라서 HTDE 기법의 작업 증가량과 모델 정확도의 트레이드 오프 관계가 존재하지만 합리적인 수준이라고 볼 수 있다.

5. 결론

본 논문은 최근 기계 독해에서 준수한 성능을 보여주는 BERT 모델을 기반으로 KorQuAD 2.0 데이터 셋을 활용하여 HTML 문서 구조의 데이터를 효과적으로 학습하기 위한 HTDE 기법을 제안했다. 실험을 통해 모델 정확도 평가 지표인 EM, F1 점수를 기준으로 측정한 결과 HTDE 기법이 절대 위치 임베딩 기법을 사용한 모델보다 각각 약 25.9~58.3%, 16.2~32% 높은 결과를 보였고, 질의 응답 문제에서 좋은 성능을 보이는 상대 위치 임베딩 기법을 사용한 모델보다도 각각 약 10.2~11.7%, 5.9~6.2% 높은 결과를 보임으로써, 절대 위치 임베딩과 상대 위치 임베딩을 사용한 경우보다 모델의 정확도를 향상시킬 수 있음을 입증했다. 그리고 학습 스텝에 따른 학습 손실과 에폭 수에 따른 평가 손실에 대한 실험을 진행한 결과 HTDE 기법의 경우 다른 두 기법보다 최초 평가 손실이 다소 높았으나 학습을 통해 최종 평가 손실이 각각 약 5.83e2~6.66e2%, 2.33~9.26% 수준으로 더 낮은 결과를 보였다. 이는 HTDE 기법에서 HTML 태그 깊이 임베딩 레이어를 추가하면서 학습되지 않은 파라미터들이 발생했기 때문에 학습 초기의 손실이 높을 수 있으나, 학습이 진행됨에 따라 HTML 문서의 구조를 통해 보다 효과적으로 문서를 학습할 수 있음을 나타낸다. 또한 임베딩 기법에 따른 FLOPS를 측정하는 실험에서 HTDE가 다른 두 임베딩과 달리 HTML 태그 데이터 처리로 인해 작업량이 약 0.716% 수준으로 소폭 증가하는 것을 확인했다. 따라서 향후에는 추가적인 작업량 증가 없이 모델의 정확도를 개선할 수 있는 임베딩 기법을 연구할 계획이다.

REFERENCES

[1] J.Lee, “Analysis of the Hardware Structures of the IoT Device Platforms for the Minimal Power Consumption” Journal of Internet of Things and Convergence, Vol.6, No.2, pp.11-18. 2020.

[2] H.S.Lee, “A Prediction-Based Data Read Ahead Policy using Decision Tree for improving the performance of

NAND flash memory based storage devices" Journal of Internet of Things and Convergence, Vol.8, No.4, pp.9-15. 2022.

[3] S.H.Lee and D.W.Lee, "A Software Engineering-Based Software Development Progress Analysis in IoT Environment" Journal of Internet of Things and Convergence, Vol.6, No.2, pp.87-92. 2020.

[4] T.Kwiatkowski, J.Palomaki, O.Redfield, M.Collins, A.Parikh, C.Alberti, D.Epstein, I.Polosukhin, J.Devlin, K.Lee, K.Toutanova, L.Jones, M.Kelcey, M.W.Chang, A.M.Dai, J.Uszkoreit, Q.Le and S.Petrov, "Natural questions: a benchmark for question answering research." Transactions of the Association for Computational Linguistics, Vol.7, pp.453-466. 2019.

[5] D.Chen, A.Fisch, J.Weston and A.Bordes, "Reading wikipedia to answer open-domain questions." Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vol.1, pp.1870-1879. 2017.

[6] E.Choi, H.He, M.Iyyer, M.Yatskar, W.Yih, Y.Choi, P.Liang and L.Zettlemoyer, "QuAC: Question answering in context." Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp.2174-2184, 2018.

[7] A.Asai, X.Yu, J.Kasai and H.Hajishirzi, "One question answering model for many languages with cross-lingual dense passage retrieval." Advances in Neural Information Processing Systems, Vol.34, pp.7547-7560, 2021.

[8] J.Devlin, M.W.Chang, K.Lee and K.Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding." Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Vol.1, pp.4171-4186, 2019.

[9] B.Wang, L.Shang, C.Lioma, X.Jiang, H.Yang, Q.Liu, and J.G.Simonsen, "On position embeddings in bert." International Conference on Learning Representations, 2020.

[10] P.Rajpurkar, J.Zhang, K.Lopyrev, and P.Liang, "Squad: 100,000+ questions for machine comprehension of text." Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp.2383-2392, 2016.

[11] Z.Yang, P.Qi, S.Zhang, Y.Bengio, W.W.Cohen, R.Salakhutdinov and C.D.Manning, "HotpotQA: A dataset for diverse, explainable multi-hop question answering." Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp.2369-2380, 2018.

[12] S.Y.Lim, M.J.Kim, and J.Y.Lee. "Korquad: Korean qa dataset for machine comprehension." Proceeding of the Conference of the Korea Information Science Society, Vol.45, No.2, pp.539-541, 2018.

[13] Y.M.Kim, S.Y.Lim, H.J.Lee, S.Y.Park and M.J.Kim, "KorQuAD 2.0: Korean QA dataset for web document machine comprehension." Proceeding of the Conference of the Korea Information Science Society, Vol.47, No.06, pp.577-586, 2020.

[14] I.Sutskever, O.Vinyals, and Q.V.Le, "Sequence to sequence learning with neural networks." In Proceedings of the 27th International Conference on Neural Information Processing Systems, Vol.2, pp.3104-3112, 2014.

[15] D.Bahdanau, K.H.Cho and Y.Bengio, "Neural machine translation by jointly learning to align and translate.", 3rd International Conference on Learning Representations, 2015.

[16] A.Vaswani, N.Shazeer, N.Parmar, J.Uszkoreit, L.Jones, A.N.Gomez, Ł.Kaiser and I.Polosukhin, "Attention is all you need." Advances in neural information processing systems, Vol.30, pp.5998-6008, 2017.

목 진 왕(Jin-Wang Mok) [준회원]

- 2016년 3월 ~ 현재 : 백석대학교 컴퓨터공학부 인공지능전공 학사과정
- 2021년 12월 ~ 현재 : 백석대학교 학부연구원

〈관심분야〉
자연어 처리, 음성 신호 처리, 이미지 처리, 블록체인

장 현 재(Hyun Jae Jang) [종신회원]

- 2013년 2월 : 고려대학교 뇌공학과 (공학 석사)
- 2017년 8월 : 고려대학교 뇌공학과 (공학 박사)
- 2017년 9월 ~ 2021년 2월 : 고려대학교 뇌공학과 연구교원
- 2021년 3월 ~ 현재 : 백석대학교 컴퓨터공학부 조교수

〈관심분야〉
인공지능, 뉴로모픽, 계산 신경망

이 현 섭(Hyun-Seob Lee) [종신회원]

- 2007년 2월 : 한양대학교 컴퓨터 공학과 (공학 석사)
- 2013년 2월 : 한양대학교 컴퓨터 공학과 (공학 박사)
- 2012년 3월 ~ 2021년 2월 : 삼성전자 책임연구원
- 2021년 3월 ~ 현재 : 백석대학교 첨단IT학부 조교수

〈관심분야〉
인공지능, 저장시스템, 임베디드 시스템