한국어 중심의 토큰-프리 언어 이해-생성 모델 사전학습 연구

신종훈○, 허정, 류지희, 이기영, 서영애, 성진, 임수종

한국전자통신연구원 언어지능연구실
{jhshin82, jeonghur, chrisjihee, leeky, yaseo, real_castle, isj}@etri.re.kr

Towards Korean-Centric Token-free Pretrained Language Model

Jong-Hun Shin○, Jeong Heo, Ji-Hee Ryu, Ki-Young Lee, Young-Ae Seo, Jin Seong, Soo-Jong Lim
Electronics and Telecommunications Research Institute, Language Intelligence Research Section

요 약

본 연구는 대부분의 언어 모델이 사용하고 있는 서브워드 토큰화 과정을 거치지 않고, 바이트 단위의 인코딩을 그대로 다룰 수 있는 토큰-프리 사전학습 언어모델에 대한 것이다. 토큰-프리 언어모델은 명시적인 미등록어 토큰이 존재하지 않고, 전 처리 과정이 단순하며 다양한 언어 및 표현 체계에 대응할 수 있는 장점이 있다. 하지만 관련 연구가 미흡, 서브워드 모델에 대비해 학습이 어렵고 낮은 성능이 보고되어 왔다. 본 연구에서는 한국어를 중심으로 토큰-프리 언어 이해-생성 모델을 사전 학습 후, 서브워드 기반 모델과 비교하여 가능성을 살펴본다. 또한, 토큰 프리 언어모델에서 지적되는 과도한 연산량을 감소시킬 수 있는 그래디언트 기반 서브워드 토크나이저를 적용, 처리 속도를 학습 2.7배, 추론 1.46배 개선하였다.

주제어: 사전학습 언어모델, 언어 이해 생성 모델, 토큰화 생략

1. 서론

사전학습 언어모델은 Transformer[1] 신경망 구조를 기반으로, 주어진 문맥의 다음 어휘를 예측하거나 입력 문맥의 일부분을 마스크(mask) 토큰을 통해 가리는 등 손상시킨 후 이를 복원하는 방법으로 입력된 문맥의 고유한 벡터 표현을 학습한다.

학습 과정에서는 입력된 문자열을 N개의 토큰으로 분리하는 과정, 즉 토큰화(Tokenization) 과정이 필요하다. 과거에는 규칙에 의한 단조 토큰화 및 형태소 분석, 음절/문자 단위로 분해하는 경우가 많았으나, 현재는 WordPiece, BPE 등과 같은 서브워드(Subword) 토큰화 모델링 방법이 성공적으로 자리잡아 현재는 사실상 표준으로 사용되고 있다. 이들 서브워드 모델링은 연속된 문자의 출현 빈도 또는 언어모델의 perplexity를 기준으로 최적화됨으로 종종 언어학적으로 부적절한 단위로 토큰화 된다는 비판[2]을 받아왔으나, 한국어 중심 모델에서는 HyperCLOVA[3], KLUE-RoBERTa[4]와 같이 형태소 분석된 말뭉치로 토큰화 모델을 학습한 형태소 인지(Morphologically-Aware) BPE 모델링으로 보완되어 왔다.

그럼에도 불구하고 명시적인 토큰화에 의해 고정된 사전은 학습 당시 말뭉치의 통계적 특성이 고착되며, 학습 당시에 다루어 지지 못한 문자를 포함한 어휘가 미등록어로 처리된다는 문제를 가지고 있다. 이 중 미등록어 문제는 바이트 수준의 BPE(Byte-level BPE)[3][5]로 해소되고 있으나, 여전히 구어체 표현과 같은 비정형 텍스트에서 발생하는 오타, 닮은꼴 표현, 음역, 이모지(emoji)를 포함해, 욕설 필터링이나 자동번역을 우회하기 위한 적대적 변형에 취약하다[2]. 또한, 의학, 법률 등 사전 학습에 참여하지 못한 도메인에 속하는 다운스트림 태스크의 낮은 성능과, 이를 해소하고자 등장하는 도메인 특화 언어모델[6] 연구에서도 고정된 토큰화 방법에 일정한 책임이 있음을 종종 지적하고 있다.

이러한 토큰화 과정을 생략하게 하고, 모델이 입력 그대로를 End-to-End 형식으로 학습하게 하는 접근을 토큰-프리(Token-free) 방법[2][7]으로 정의한다. 토큰 프리 접근 방법은 사전학습 모델의 학습 단계에서 제외된 미등록 어휘가 존재하지 않게 만드는 장점이 있고, 복잡한 입력 전 처리 단계에서 발생하는 인적 오류에 상대적으로 자유로울 뿐만 아니라 정규화 시점에서 발생하는 문맥 정보의 손실도 최소화할 수 있다. 또한, 다른 문자 체계를 사용하는 언어와 혼합 가능성, 음성, 시각 모달리티와의 결합에도 입력 수준에서의 정렬 단위가 어긋나지 않는 등 다양한 응용에 적합하게 만들 수 있는 가능성을 가지고 있다. 그럼에도 불구하고, 아직 이에 관련한 시도가 많지 않아, 단일언어 대상 벤치마크에서는 서브워드 기반 언어 모델에 비해 학습이 어렵고 상대적으로 낮은 성능[8]이 보고되어 왔다. 또한, 더 작은 단위의 토큰은 필연적으로 같은 문맥을 표현하기 위해 더 많은 시퀀스 길이를 요구하게 되고, 학습과 추론에 더 많은 연산을 필요로 한다는 단점이 있다.

본 연구에서는 입력을 토큰화 하지 않고, 바이트 표현만으로 구성된 사전과 임베딩을 사용하는 토큰-프리 언어모델을 한국어 중심으로 구축하고, 이를 서브워드 기반 모델과 비교, 경쟁력을 가지고 있는지를 확인하고자 한다. 또한, 증가한 연산량을 효과적으로 낮추는 그래디언트 기반 서브워드 토큰화 기법을 적용하여, 실용적인 토큰-프리 언어모델에 대한 가능성을 보이고자 한다.

2. 관련 연구

2.1 바이트 단위 언어 이해-생성 모델 연구

ByT5[7]는 바이트 수준의 입출력을 다루는 언어 이해생성모델로, 10억 이하의 파라미터를 갖는 다국어 언어 이해생성 모델 mT5의 단점을 보완하기 위해 제안되었다. 약 25만개의 어휘를 사용하는 mT5는 10억 파라미터 이하의 모델에서 어휘 임베딩이 전체 파라미터의 50% 이상을 차지하게 됨으로 파라미터 부족에 의한 성능저하 문제가 발생하는데, ByT5에서는 입력 임베딩을 바이트 단위로 구성하는 방법으로 이를 완화한다. 시퀀스 길이가 길어지므로 인해 발생하는 처리/생성속도의 저하는 디코더 계층을 인코더 계층과 비대칭적으로 구성, 3:1 비율로 생성하여, 디코딩 속도를 개선했다. 선행 연구에서는 문자 수준의 노이즈를 가한 평가 데이터를 통해 서브워드 모델 대비 강건함을 보였으나, 벤치마크 수준의 깨끗한 입력으로 구성된 테스트에서는 10억 미만 규모의 모델을 제외하면 동등한 규모의 mT5 모델 대비 성능 개선이 제한적이거나 더 낮게 나타나며, 5억 이하의 파라미터를 갖는 소규모 단일언어 T5 모델에 비해서도 낮은 성능이 나타나는 등 개선의 여지가 남아있는 결과를 보였다.

ByT5 모델은 문자 단위, 또는 바이트 단위의 사전학습 언어모델의 가능성을 보였지만, 그것의 한계 또한 여실히 보여주고 있다. 더 작은 단위의 토큰 표현은 입출력 노이즈로부터 강건한 면모를 보이나, 긴 길이의 입출력을 다루는 현대의 자연언어처리 응용에서는 같은 표현을 생성하더라도 시퀀스 길이가 길어 짐으로 더 많은 연산량과 높은 지연시간을 가지게 된다.

2.2 그래디언트 기반 서브워드 토크나이저

CharFormer[9]는 앞서 ByT5에서 보인 바이트 단위의 연산이 인코더에 미치는 비효율성을 개선하기 위해 그래디언트 기반의 서브워드 토크나이저(Gradient-based Subword Tokenizer; 이하 GBST)를 제안하였다.

End-to-End 형식의 서브워드 학습을 위한 GBST 구현은 다음과 같다. 먼저 바이트 단위로 임베딩 표현이 부여된 입력 X에 대해, 기 지정된 후보 블록 길이 집합 b에 따라, 1D-컨볼루션을 사용해 후보 서브워드 블록 $\mathrm{X_{b,i}} \in \mathbb{R}$을 생성하고, 이를 단순 선형변환 $\mathrm{F_R}: \mathbb{R}^d \rightarrow \mathbb{R}$을 통해 각 후보 블록의 점수 $\mathrm{p_{b,i}} = F_R(X_{b,i})$를 계산한다. 이 때, 토큰의 최소 단위마다 최선의 서브워드 블록을 결정할 수 있도록, 블록의 길이 b만큼 중복시켜 후보 블록의 정렬을 일정하게 만든다. 다음으로, 정렬된 후보 블록들에 대해, 학습 과정에서 최적의 후보가 크게 반영될 수 있도록 가중치 합으로 만든다. 즉, 각 위치 i에 대한 최적 스코어 $\mathrm{P_i}$와, 이에 기반한 잠재 서브워드 표현 벡터 $\hat{X_i}$는 아래의 수식으로 계산된다:

$$\mathrm{P_i} = softmax\left(\left[p_{1,i}, p_{1,i}, \ldots p_{M,i}\right]\right)$$

$$\hat{X_i} = \sum_{b}^{M} P_{b,i} X_{b,i}$$

GBST 레이어는 과거 문자 기반의 언어모델 연구에서 종종 차용된 컨볼루션 기반의 어절 정보 결합 메커니즘과 그 결이 유사하다. Pooling을 통해 자질을 취합하는 대신, 가중치 합의 꼴로 만들어냈다는 점, 그리고 이를 통해 생성한 자질을 원본의 정보와 concatenate하여 자질 강화 목적으로 어절 정보를 만들어냈던 것과 달리, 순수하게 변환된 블록 별 가중치 합에만 의존한다는 점에서 차이가 있다. 다만, End-to-End 형식으로 서브워드 정보로 치환하였음에도 불구하고, 근본적으로 바이트 단위를 다룬다는 점에서 과도한 연산량의 요구는 해소되지 않는다. 이를 완화하기 위해, 생성된 잠재 서브워드 표현을 상위의 트랜스포머 스택으로 전달하기 전에 1/k 길이(예: k=3)로 만들도록 Mean Pooling을 수행, 다운샘플링(downsampling) 하여 연산량을 효과적으로 절감한다.

3. 토큰프리 사전학습 언어 모델 구현의 상세

본 연구에서는 한국어 중심의 ByT5 모델을 기본 구조로 학습 후, GBST 계층을 인코더에 결합한 뒤, 기반 모델의 가중치를 유지하고, 신규 계층만 초기화 하여 추가 학습하는 기법(uptraining)을 통해 학습하였다. 학습 대상의 규모, 구성은 아래의 표와 같다:

타입	#params	L_{enc}	L_{dec}	D_{ff}	D_{model}
Small	330M	12	4	3584	1472
Base	580M	18	6	3968	1536
Large	1.23B	36	12	3840	1536

3.1 바이트 단위 언어 이해-생성 모델(ByT5)

초기에는 상기 3개 규모보다 작은 소형 모델(~130M #params)급도 학습에 포함하였으나, 성능 격차가 커 대상에서 제외되었으며, 평가 역시 Base와 Large 모델로 한정했다.

3개 타입은 모두 Hugging Face Hub에 공개된 ByT5 모델(google/byt5)의 가중치로 초기화 하였는데, 이는 ByT5 모델의 토큰 임베딩 및 사전의 수가 384개(3개의 예약 토큰 + 256 bytes + 125개의 sentinel token; 이 중 125는 TPU의 메모리 정렬을 위해 추가됨)에서 변화될 필요가 없기 때문이다. NIA의 AIHub 한국어 텍스트 데이터셋, 국립국어원 모두의 말뭉치, 한국어 위키 덤프, 웹 뉴스 등으로 구성된 ~220GB 수준의 학습 데이터를 사용했고, 통상의 T5 학습에는 1M 스텝을 학습하는데, 본 연구에서는 절반에 못 미치는 약 400k 스텝 정도로 학습 후 평가되었다. 유효 배치 크기는 1M token(=bytes)으로 설정하였으며, 초기에는 4장의 A100 80GB 단일 장비로, 중/후기에는 8장의 A100 80GB GPU로 구성된 단일 장비를 사용하여 학습하였다. 정밀도는 bfloat16을 사용하여 학습하였으며, 옵티마이저는 AdamW와 Inverse-Sqrt 학습률 스케쥴러를 사용(초기 LR=1e-4, Warmup=8000)하여 학습을 시작했다. 통상의 T5 계열 모델 학습 요령과 달리, 초기 테스트에서 Adafactor 옵티마이저 사용 간에 학습 불안정이 관측, AdamW가 사용되었다.

3.2 GBST가 적용된 바이트 단위 T5 모델

GBST의 적용은 앞서 기술한 바와 같이, 학습된 가중치를 재사용하여 학습되는데, uptraining을 통해서 모델이 변환될 때 각 모델마다 약 4백만개(4M)의 파라미터가 추가된다. 한국어의 UTF-8 바이트 표현은 통상 3바이트가 1개의 음절 표현에 사용되고 있으며, 통상의 3만단어급 서브워드 모델의 평균 서브워드 음절 길이는 약 2.3단어로, 이는 약 7 bytes에 해당, 이를 기준으로 서브워드 후보 블록의 크기를 [1, 2, 3, 6, 9]로 설정하였다. 다운샘플링 팩터는 3으로 설정하였다. 즉, GBST 레이어를 거쳐 트랜스포머 인코더 스택에 전달되는 입력의 길이는 서브워드 블록 크기의 공배수 길이만큼 패딩(padding)된 뒤, 1/3 크기로 줄어든다. 학습에는 같은 말뭉치를 사용했고, 3.1에서 학습된 스텝 수의 절반 수준으로 학습하고자 하였으나, 가용장비 부족의 문제, 성능 확보 과정에서 학습량이 정확하게 통제되지 못하는 어려움이 있었다. 23년 9월 현재 모든 모델의 학습이 완료되지 않아, 하위 태스크 성능 실험에서는 Base급만 사용하였다.

3.3 학습 및 마스킹 전략

선행연구[7]를 그대로 답습하는 경우에는 서브워드 기반 모델 대비 낮은 성능이 기대될 것이 확연하므로, 학습과정을 개선하여 토큰-프리 모델의 성능 확보를 시도하였다. 통상적으로 노이즈 밀도를 15%로 설정하는 것에 반해, 최소 20%에서 최대 40%까지 노이즈 밀도를 증가시키며 학습하였다. 마스킹은 ByT5 학습에 사용한 바이트 수준의 랜덤 마스킹을 변형하여, 유니코드 음절 단위로 마스킹 Span을 형성하게 한 뒤, Span Denoising 학습을 수행했다. Span의 평균 길이는 한국어 9음절에 해당하는 27 bytes로 책정하였다. 학습 간에는 장비 부족 문제를 완화하기 위해 입력 문맥의 길이를 증가시키는 커리큘럼 학습을 수행, 단위는 512, 1024, 2048 바이트로 증가시켰다. 학습 데이터는 통상의 한국어 대상 언어모델 학습 시 입력을 문장 단위로 자르는 방법과 달리, 입력 길이에 맞게 단순 분할하는 방법을 사용하여 학습하였다.

본 연구에서는 비록 다루지 못하였으나, 차후 연구에서는 문장단위 표현의 학습 개선을 포함해, Span의 길이를 극단적으로 늘리거나, Prefix LM 학습 태스크 등 다양한 사전학습 목표를 추가하여 보강 학습할 예정이다.

4. 하위 태스크 실험 및 평가

학습된 언어모델의 성능 평가를 위해 KLUE[4] 벤치마크를 사용하였다. 해당 데이터셋은 언어 이해모델의 평가를 위해 주로 사용되나, 공개된 평가 말뭉치로 쉬운 재현이 가능하고, 기초적인 학습 능력 파악에는 무리가 없을 것으로 판단하여 이를 선택했다. 학습 데이터의 5%를 샘플링하여 검증 데이터셋으로, 원래 제공된 검증 데이터셋(=dev set)을 사용하여 평가를 실시했다. 본 논문에서 사용된 하위 태스크는 Seq-to-seq 스타일로 변형되어 미세조정 단계를 거쳤다. 미세조정을 위해 모든 평가 태스크는 4 epoch 만큼 학습되었으며, 배치 크기 16, 학습률은 Base 모델 및 비교군에 8e-5, Large 모델에 4e-5를 설정하여 별도의 warmup step이 없이 학습하였다. 학습률 스케쥴러로는 Cosine Annealing[10] 기법을 사용하였으며, 최소 학습률(eta_min)=1e-7, 매 epoch마다 리셋, 리셋 시 감가율(gamma)=0.7로 설정하였다. 평가셋의 추론은 greedy한 방법으로, 별도의 샘플링 없이 태스크별로 예상되는 최대 길이를 설정(<6000 bytes)하여 End-of-Sentence 토큰이 나올 때까지 생성하였다.

테스트에 사용된 모델은 학습이 일정수준 완료된 모델을 대상으로, GBST가 적용되지 않은 토큰-프리 모델 2종(KEByT5-Base, KEByT5-Large), 그리고 GBST가 적용된 토큰 프리 모델 1종(GBST-KEByT5-Base)을 사용했다.

비교군은 동등 구조를 갖는 바이트 단위 언어 이해-생성 모델인 Google/ByT5-Large (1.23B #params)[7]와, BBPE 서브워드 토큰화 모델이 적용된 paust/pko-t5-large (800M #params)[11]를 사용하였다. 이 때, 측정 기준과 입출력 구성에 차이가 없는 일부 태스크에 대해서는 편의상 [11]에서 보고된 수치를 그대로 사용하였다. 첫번째 비교군을 통해 한국어 말뭉치로 언어 적응 사전학습(Language-Adaptation Pre-Training)후 성능 개선을 살펴보고, 두번째 비교군을 통해서 서브워드 모델과 비교해 토큰-프리 모델의 경쟁력을 확인하였다. 또한, 언어 이해모델로 구조가 달라 직접 비교에는 부적합하나, 검증 데이터셋의 성능이 기재된 KLUE-RoBERTa-Large 모델[4] (337M #params)을 참고 수치로 추가, 이탤릭 체로 기재하였다.

4.1 KLUE-TC(YNAT) 분류 테스트

Seq-to-seq 모델을 사용한 분류 태스크는 신문 기사의 제목만 사용하여 입력 컨텍스트를 구성하였고, 출력 레이블 7개의 텍스트 표현(예: IT과학, 경제, 사회, …)을 별도의 데이터 증강 없이 정답으로 학습, 평가하였다.

모델	Macro Avg. F1
KLUE-RoBERTa-Large[4]	*85.88*
Google/ByT5-Large (1.23B)	78.52
paust/pko-t5-large (800M)[11]	**87.12**
KEByT5-Base (580M)	84.99
KEByT5-Large (1.23B)	85.68
GBST-KEByT5-Base (584M)	85.29

KLUE-TC(YNAT) 평가 결과에서는 다국어 모델인 Google/byt5-large 대비 적은 파라미터 규모에서도 더 나은 성능을 보였고, 언어 이해모델과도 경쟁 가능한 수준의 결과를 볼 수 있다. 하지만 동등 메커니즘을 사용하는 최신의 한국어 서브워드 모델 비교군의 보고 수치[11] 대비 다소 낮은 성능을 보인다. 동등한 사전학습 말뭉치를 사용하지 않았고, 문장 수준의 사전학습이 이루어지지 않았던 점을 고려, ~50 글자 정도의 짧은 문맥의 표현 학습에 대해서는 추가 분석과 개선이 요구된다. 한편, GBST의 출력이 1/3로 줄어드는 다운샘플링이 적용되었음에도 불구하고, 성능이 개선되었음을 확인할 수 있었다.

4.2 KLUE-NER(개체명 인식) 테스트

NER을 위한 입력 컨텍스트는 원시 문장만을 사용하였고, 출력은 입력 컨텍스트에 주어진 원시 문장을 그대로 생성하되, 검출된 개체명을 대괄호 기호를 사용해 태그 형식과 개체명 클래스 레이블을 함께 부착하여 결과물을 생성하도록 구성하였다.

평가를 위해, 생성된 원문+태그 혼합 결과에서 태그로 감싸진 문자열만 추출 후 평가했다. KLUE에서 사용한 Entity 수준 F1은 언어 이해모델 기준의 베이스라인 산출 방법과 동일하게 산출할 수 있으나, Char F1은 이해생성 모델의 특성상 입/출력의 길이가 서로 상이하게 나타나기 때문에, 기존 산출 방법은 IOB2로 표현된 시퀀스에서 'O' 를 제외한 나머지의 매크로 평균 F1을 산출하는 것이나, 본 실험에서는 엔티티를 구성하는 글자의 수만큼 태그로 치환, (예: <원빈:PS> -> PS PS) 남은 태그 시퀀스에 대한 Word-level Unigram F1을 chrF[12]로 산출하여 가능한 유사한 결과가 나올 수 있게 하였다. 결과는 아래와 같다:

모델	Entity F1	Char F1
KLUE-RoBERTa-Large[4]	*84.54*	*91.45*
Google/ByT5-Large(1.23B)	48.81	63.95
paust/pko-t5-large(800M)	**89.49**	**93.34**
KEByT5-Base(580M)	86.75	91.05
KEByT5-Large(1.23B)	88.09	92.40
GBST-KEByT5-Base(584M)	87.35	92.09

평가 결과, NER 태스크에서도 서브워드 기반 대조군에 비해 다소 낮지만, 비교 가능한 수준의 성능을 보이고 있다. 참고로, 서브워드 기반 대조군은 [11]에서 보고된 Entity-F1 수치 88.18보다 높게 나타났다. 본 실험에서도, 동일하게 GBST 적용에 의한 성능 향상이 관측되었다.

한편, 원문을 함께 생성하는 방법을 적용했을 때, 입력의 길이보다 적게 생성하는 경우는 없었다, 하지만 과대 생성, 특히 특정 어구의 반복(예: 'ㅋ')으로 입력이 구성된 경우 이후에 등장할 NER 태그를 생성하지 못하고 기 지정된 최대길이에 도달하는 경우가 존재했다. 이렇게 오 생성된 샘플은 테스트에 사용된 devset 중 3건(0.06%)을 차지하며, 토큰-프리 모델 3종 및 비교군인 pko-t5-large에서도 동일하게 발생하였다.

4.3 KLUE-DP(의존구조 파싱) 테스트

일반적으로 의존 파싱을 위해서는 의존관계 상 헤드의 번호를 예측하고, 두 어절 사이의 의존 레이블을 예측한다. 통상적으로 언어 이해모델과 함께 동기화 된 Seq-to-seq 구조나 스택 구조 등의 보조 신경망을 함께 구현한다. 이와 달리, 본 연구에서는 의존 파싱 태스크를 위한 입력 컨텍스트와 출력 형식을 아래와 같은 구조로 구성, Auto-Regressive 디코딩에서 한번의 시퀀스 생성으로 헤드 레이블 및 의존관계 레이블을 모두 예측하도록 하였다. 즉, 입력은 원문 + 개행문자(CR) + (어절 인덱스, 어절, 어절의 문자 길이, 형태소, 품사) (…,)의 구성을 취한다. 괄호로 표현된 어절 정보는 주어진 어절 수에 맞게 삽입되고, 어절 내 문자의 길이를 제외한 나머지는 데이터셋에 주어진 값을 그대로 사용한다. 출력의 경우, (어절 인덱스/어절 길이, 헤드 번호, 의존 레이블), (…,) 와 같은 형식으로 구성하였다.

어절 단위와 동일한 개수의 예측을 수행하는 하위 태스크를 seq-to-seq으로 해결하는 데는 항상 개수 불일치의 문제가 있다. 특히, 과소생성(under-generation)의 경우에는 미처 예측되지 못한 어절이 중간에 포함되어 있을 때, OMR 카드의 답지가 밀리는 것과 같이 다른 부분의 이후가 모두 틀렸다고 표기되어야 하기 때문에, 과소생성을 최소화하는 것이 문제 해결의 핵심이 된다. 몇 가지 테스트와 변조를 거쳐, 각 어절 별 문자 수를 예측하게 하는 방법을 적용함으로, 과소 및 과대생성을 완화할 수 있었다.

본 의존 파싱의 seq-to-seq 평가는 통상의 의존 파싱 평가 방법을 그대로 사용했다. 즉, 파싱 실패, 과소생성으로 인한 오류를 모두 틀린 지점 이후를 정렬하지 않고, 틀린 것으로 처리하였으며, LAS가 틀린 경우 UAS 레이블을 무조건 오답으로 처리하는 방법으로 측정했다.

동일한 추론 조건에서 생성된 결과물 중, KEByT5-Large 모델에서 파싱 실패 1건(전체의 0.02%), 과대 생성 2건이 발생했고, GBST-KEByT5-base 모델에서는 파싱 실패는 0건, 과소생성 1건, 과대 생성 1건이 발생하였다. 비교군으로 작업한 paust/pko-t5-large 모델에서는 파싱 실패 0건, 과소생성 1건이 발생하였다. Google/ByT5-Large의 경우, 파싱 실패가 737건(전체의 14.74%), 과소 생성이 139건, 과대 생성 1,341건이 발생하였다.

실험 결과는 아래와 같다:

모델	UAS	LAS
KLUE-RoBERTa-Large[4]	*93.84*	*87.93*
Google/ByT5-Large(1.23B)	44.26	7.805
paust/pko-t5-large(800M)	85.73	84.67
KEByT5-Base(580M)	**88.70**	**85.90**
KEByT5-Large(1.23B)	87.18	85.52
GBST-KEByT5-Base(584M)	88.33	85.00

본 실험을 통해, 동등한 규모의 1/3로 다운샘플링 된 GBST 모델의 성능과 적용되지 않은 바이트 단위 모델에서 서로 비교 가능한 성능이 나왔음을 확인할 수 있었으며, 더 큰 규모의 서브워드 모델 대비해서도 나은 성능을 모든 테스트 대상 모델로부터 확인할 수 있었다.

한편, 성능 평가를 위해 위와 같은 구조로 의존 구조 추론을 실시했으나, 도메인 외 입력에 대해서는 토큰 프리 여부와 상관없이 파싱 실패 및 과소/과대 생성이 일어날 수 있으므로, 생성 언어모델 기반 응용이 점차 증가하는 지금, 입-출력의 수를 잘 지킬 수 있는 프롬프트 가이드에 대한 연구가 요구된다.

4.4 KLUE-MRC(기계 독해) 테스트

KLUE-MRC 기계 독해 태스크에 대해서도 학습 능력을 테스트하였다. 기계 독해 문제는 추출식(extractive)으로 접근하는 언어 이해 모델에 적합하게 디자인되어 있

어, 생성 및 이해생성 모델에서는 정확히 같은 구간과 표현을 따르지 못하는 경우 등 상대적으로 낮은 성능을 보인다. google/byt5-large 모델은 테스트에서 제외하였다.

MRC 태스크를 위한 입력 구성은 질문과 지문을 결합하는 통상적 처리를 수행했다. 한편, KLUE-MRC는 SQuAD 2.0과 같이 정답이 없는 경우를 포함하고 있어, 답이 없는 경우 [답이 없다]고 출력하도록 학습했다. 성능 개선을 위해, 학습 데이터셋을 1.5배 오버샘플링 후, 노이즈를 포함한 정답 또는 거짓(plausible) 정답을 부여하고, 생성 시퀀스의 뒤에서 앞의 예측을 정정하도록 입출력을 정의하고, 평가 시 시퀀스의 마지막에 표현된 정정 답안을 선택하는 방법으로 성능을 개선했다.

모델	EM	ROUGE-W
KLUE-RoBERTa-Large[4]	*75.26*	*80.30*
paust/pko-t5-large(800M)	67.49	73.48
KEByT5-Base(580M)	62.28	68.38
KEByT5-Large(1.23B)	**70.07**	**75.81**
GBST-KEByT5-Base(584M)	59.69	66.44

비교군의 서브워드 모델 재현 실험에서는 기존 공개된 수치(EM 68.01) 대비 약간 낮게 나타났다. Large 규모의 토큰-프리 언어모델은 서브워드 모델 대비 더 나은 성능을 보일 수 있었으나, 상대적으로 작은 파라미터 규모(580M)의 모델에서는 GBST 적용 여부와 상관없이 크게 못 미치는 성능이 나타났다.

4.5 GBST 적용을 통한 학습/추론 효율성 테스트

A100 80GB PCIE, 학습에는 8장, 추론에는 1장을 사용, KLUE-MRC 태스크 학습 1 epoch(3,292 배치)의 시간과 추론 시간을 측정하였다. 정밀도는 bfloat16를 사용했다.

모델	학습 샘플/초	추론 샘플/초
paust/pko-t5-large (800M)	2.17	5.12
KEByT5-base (580M)	1.30	3.95
GBST-KEByT5-base (584M)	**3.56**	**5.77**

위와 같이, GBST를 적용함으로 학습에 2.7배 이상, 추론에 1.46배 이상 개선되었으며, 서브워드 모델에 비해 파라미터 수가 다소 적으나 긴 시퀀스가 큰 페널티로 작용하지 않음을 확인할 수 있다.

5. 결론

본 연구를 통해 학습된 토큰-프리 사전학습 언어모델을 사용하여, 언어 이해 능력 검증에 사용되는 KLUE 벤치마크 데이터셋을 통해 단순 분류 및 NER 태스크에서 서브워드 모델 대비 다소 낮은 성능을 보였으나 의존구조 파싱 및 기계 독해 태스크에서 서브워드 모델보다 우수함을 보임으로, 비교가능한 성능을 가질 수 있음을 확인하였으며, 성능 및 학습 속도 비교를 통해 GBST를 적용한 토큰-프리 모델의 경쟁력을 확인할 수 있었다.

추후에는 다양한 사전학습 전략을 통해 성능 개선을 모색하고, 다국어 번역 및 적대적 텍스트 변조, 크로스 모달 결합 등 토큰-프리 모델의 강점을 비교 평가할 수 있는 연구로 확장하고자 한다.

감사의 글

이 논문은 2022년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. RS-2022-00187238, 효율적 사전학습이 가능한 한국어 대형 언어모델 사전학습 기술 개발).

참고문헌

[1] Vaswani et al., “Attention is All you Need”, in Procs. of the NeurIPS 2017. 2017.

[2] Clark et al., “CANINE: Pre-Training an Efficient Tokenization-Free Encoder for Language Representation”. TACL2022; 10 73-91. 2022.

[3] Kim et al. “What Changes Can Large-scale Language Models Bring? Intensive Study on HyperCLOVA: Billions-scale Korean Generative Pretrained Transformers”, EMNLP2021, 2021.

[4] Park et al., “KLUE: Korean Language Understanding Evaluation”, NeurIPS 2021 Track Datasets and Benchmarks (Round 2). 2021.

[5] Wang et al., “Neural machine translation with byte-level subwords”, in Procs. of the AAAI. Vol.34. No.05. 2020.

[6] Kim et al., “A pre-trained BERT for Korean medical natural language processing”, Sci Rep 12, 13847. https://doi.org/10.1038/s41598-022-17806-8. 2022.

[7] Xue et al., “ByT5: Towards a Token-Free Future with Pre-trained Byte-to-Byte Models”, TACL Volume 10, MIT Press. Pp. 291-306, 2022.

[8] Choe et al., “Bridging the Gap for Tokenizer-Free Language Models”, arXiv preprint:1908.10322. 2019.

[9] Tay et al., "Charformer: Fast Character Transformer via Gradient-based Subword Tokenization", ICLR 2022, 2022.

[10] Loshchilov and Hutter, “SGDR: Stochastic Gradient Descent with Warm Restarts”, ICLR 2017, 2017.

[11] D. Park, “pko-t5: PAUST Korean T5 for text-to-text unified framework”, https://github.com/paust-team/pko-t5. 2022.

[12] M. Popovic, “chrF: character n-gram F-score for automatic MT evaluation”, WMT, pp.392-395. 2015.