임베디드 시스템(Raspberry PI 5) 환경에서의 DistilBERT 구현 및 성능 검증에 관한 연구

임채우[1], 김은호[1], 서장원[2]
[1]동서울대학교 컴퓨터소프트웨어학과 학부생
[2]동서울대학교 컴퓨터소프트웨어학과 교수
dlacodn456@naver.com, ho1582@naver.com, jwsuh@du.ac.kr

A Study on the Implementation and Performance Verification of DistilBERT in an Embedded System(Raspberry PI 5) Environment

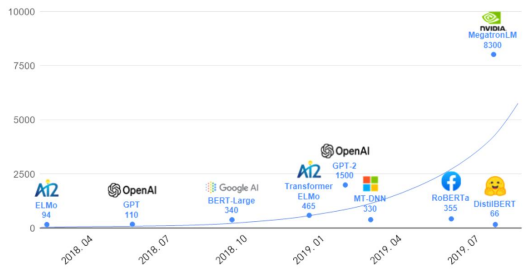
Chae-woo Im[1], Eun-Ho Kim[1], Jang-Won Suh[1]
[1]Dept. of Computer Software, Dongseoul University

요 약

본 논문에서 핵심적으로 연구할 내용은 기존 논문에서 소개된 BERT-base 모델의 경량화 버전인 DistilBERT 모델을 임베디드 시스템(Raspberry PI 5) 환경에 탑재 및 구현하는 것이다. 또한, 본 논문에서는 임베디드 시스템(Raspberry PI 5) 환경에 탑재한 DistilBERT 모델과 BERT-base 모델 간의 성능 비교를 수행하였다. 성능 평가에 사용한 데이터셋은 SQuAD(Standford Question Answering Dataset)로 질의응답 태스크에 대한 데이터셋이며, 성능 검증 지표로는 EM(Exact Match) Score와 F1 Score 그리고 추론시간을 사용하였다. 실험 결과를 통해 DistilBERT와 같은 경량화 모델이 임베디드 시스템(Raspberry PI 5)과 같은 환경에서 온 디바이스 AI(On-Device AI)로 잘 작동함을 증명하였다.

1. 서론

기존의 NLP(Natural Language Processing)는 사전 학습(Pre-Trained) 모델을 다운스트림 태스크(Downstream Task)에 전이 학습(Transfer-Learning)을 하는 것이 일반적인 방식이었다. 또한, 사전 학습 모델의 성능을 향상시키면 다운스트림 태스크에 대한 성능도 향상되었지만, 파라미터 수도 동시에 증가하였다. 이로 인해, 사전 학습 모델의 훈련 시간이 길어지고, 환경 비용도 증가하는 문제가 발생하였다. 이런 문제를 해결하기 위해 모델의 경량화 연구가 진행되었고 BERT-base 모델에서 파라미터 수는 40% 감소하고, 처리속도는 60% 증가한 DistilBERT 모델에 대한 연구가 발표되었다[1]. 언어 모델의 출시 시기와 파라미터 수의 증가 추이는 다음의 (그림 1)과 같다.



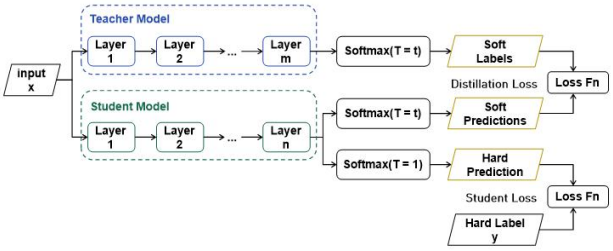
(그림 1) 언어 모델의 출시 시기와 파라미터 수

2. DistilBERT 구조

DistilBERT의 구조의 변화는 크게 3가지가 있다. 첫 번째는 Transformer Layer의 수를 절반으로 감소시켰다. 딥러닝 프레임워크 연산이 최적화되어 Layer의 수를 감소시켜도 성능 저하가 발생하지 않았다. 두 번째는 BERT-base의 구조에서 다음 문장 예측(Next Sentence Prediction)을 담당하는 토큰형 임베딩(Token-type Embedding)을 제거하였다. 토큰형 임베딩을 제거하여도 성능 저하는 미미하였다. 세 번째는 풀러 레이어(Pooler Layer)를 제거하였다. 모델을 가볍게 만들기 위해 모델에서 풀러 레이어를 제거하고, 풀러 레이어가 필요한 태스크는 미세 조정하는 과정에 풀러 레이어를 추가하여 사용할 수 있도록 하였다. 위와 같은 과정을 통하여 모델의 전반적인 시간 복잡도가 감소함에 따라 경량화가 진행되었다.

3. 지식 증류

DistilBERT의 지식 증류(Knowledge Distillation)는 모델의 경량화 기법 중 하나로서, 크고 깊은 교사 모델(Teacher Model)의 지식을 작고 얇은 학생 모델(Student Model)에게 전수하는 기법이다. 사용하는 손실 함수(Loss Function)로는 Distillation Loss와 Student Loss가 있으며, Distillation Loss는 교사 모델의 지식을 최대한 학습하는 Loss이고, Student Loss는 학생 모델의 정답과 실제 정답을 비교하여 실제 정답에 더 가깝게 예측하게 하는 Loss이다. 지식 증류의 구조도는 다음의 (그림 2)와 같다[2].



(그림 2) 지식 증류 구조도

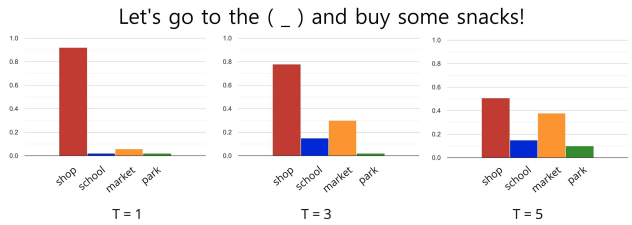
4. 소프트맥스-온도

지식 증류의 활성화 함수(Activation Function)로 소프트맥스(Softmax)와 소프트맥스-온도(Softmax-Temperature)

함수를 사용하였다. 여기서 온도 T는 상수로, 출력을 T 값으로 나눈 다음 소프트맥스 함수를 적용한 것이다. 다음의 수식 (1)은 소프트맥스-온도 함수의 수식이다.

$$q_i = \frac{\exp(z_i / T)}{\Sigma_j \exp(z_j / T)} \quad (1)$$

수식 (1)에서 T 값이 커지면 커질수록, 1순위 클래스(class)의 확률은 낮아지고 2순위 이하 클래스의 확률은 전반적으로 높아진다. 즉, 단순히 가장 높은 확률을 가진 클래스만이 아니라, 모든 클래스에 대한 확률 분포를 고려하여 학생 모델을 훈련시키는 것을 의미한다. T 값에 따른 확률 분포의 변화는 다음의 (그림 3)과 같다[3]. T=1일 때는 1순위 클래스에만 높은 확률이 부여되지만, T=3 이상일 때는 2순위 이하 클래스에도 확률이 부여되므로 모델이 모든 클래스에 대하여 학습할 수 있게 된다.



(그림 3) T 값의 변화에 따른 확률 분포의 변화

5. DistilBERT의 3가지 손실 함수

DistilBERT 모델은 지식 증류의 손실 함수를 Distillation Loss, Masked Language Modeling Loss, Cosine Embedding Loss의 3가지 손실 함수를 조합하여 사용한다. Masked Language Modeling Loss는 Student Loss와 내용은 동일하며 명칭만 변경되었다. Cosine Embedding Loss는 학생 모델의 Hidden States Vectors의 방향을 교사 모델의 Hidden States Vectors에 맞춰주는 Loss이다. 위 3가지의 Loss에 비중을 정하는 합이 1인 하이퍼 파라미터 α, β, γ를 곱한 다음 전부 더하여 수식 (2)와 같이 3가지의 손실 함수의 조합이 된다. 수식 (3)은 Distillation Loss, 수식 (4)는 Masked Language Modeling Loss, 수식 (5)는 Cosine Embedding Loss를 나타낸다.

$$L_{triple} = \alpha \times L_{distil} + \beta \times L_{MLM} + \gamma \times L_{\cos} \quad (2)$$

$$L_{distil} = (\sigma(\frac{Z_s}{T}), \sigma(\frac{Z_t}{T})) \quad (3)$$

$$L_{mlm} = (\sigma(Z_s), \hat{y}) \quad (4)$$

$$L_{\cos} = 1 - \cos(x, y) \quad (5)$$

6. 임베디드 시스템(Raspberry PI 5)

본 논문에서는 임베디드 시스템(Raspberry PI 5)에 DistilBERT 모델을 탑재하여 구현하였다. 임베디드 시스템(Raspberry PI 5)의 구현 환경은 다음의 <표 1>과 같다.

<표 1> 임베디드 시스템(Raspberry PI 5)의 구현 환경

항목		내용
H/W	CPU	BCM2712 (2.4GHz)
	GPU	VideoCore VII (800MHz)
	MEMORY	SDRAM 4267
	SD card	micro 카드 슬롯, SDR104 고속 모드 지원
S/W	O/S	Debian GNU/Linux 12
	Library	Tensorflow=2.15.0, Transformers=4.38.1, Datasets=2.18.0

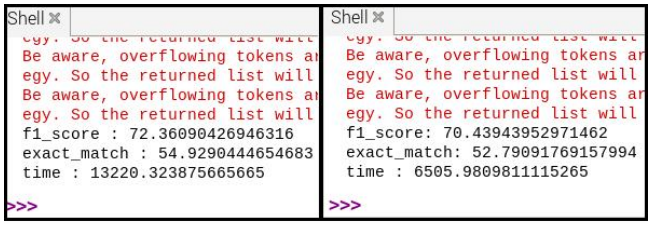
7. 구현 및 성능 비교

성능 비교 실험은 임베디드 시스템(Raspberry PI 5)에서 3개의 라이브러리를 설치하고 SQuAD 데이터셋을 사용하여 질의응답 태스크에 대하여 BERT-base 모델과 DistilBERT 모델의 성능 평가를 진행하였다. 성능 측정 지표로는 EM Score, F1 Score, Inference Time을 사용하였다. 결과는 다음의 <표 2>와 같다.

<표 2> 성능 측정 결과

Score / Model	EM Score	F1 Score	Inf. time (second)
BERT-base	54.93	72.36	13220.32
DistilBERT	52.79	70.44	6505.98

각 모델별 임베디드 시스템(Raspberry PI 5)에서 수행한 쉘의 동작화면은 다음의 (그림 4)와 같다.



(a)BERT-base (b)DistilBERT

(그림 4) 추론 속도 측정 결과 화면

실험 결과, BERT-base 모델의 EM Score는 54.93점, F1 Score는 72.36점을 기록하였다. DistilBERT 모델은 BERT-base 모델에 비해 EM Score는 4% 낮은 52.79점, F1 score는 3% 낮은 70.44점을 기록하였다. 하지만, 추론 속도는 DistilBERT 모델이 BERT-base 모델에 비해서 50% 더 빠른 결과를 얻었다.

8. 결론

본 논문에서 사용한 DistilBERT 모델은 BERT-base 모델의 경량화 버전으로 BERT-base 모델에 비해 상대적으로 적은 파라미터 수를 가지고 있다. 그러므로 임베디드 시스템(Raspberry PI 5)의 온 디바이스 AI로 적용이 가능한 모델이다. 실제 본 논문에서 사용한 임베디드 시스템(Raspberry PI 5)에서 DistilBERT 모델과 BERT-base 모델을 비교해 보았을 때, 성능 측면에서는 차이가 미미했으며 추론 속도 측면에서는 BERT-base 모델보다 DistilBERT 모델이 50% 가량 우수함이 검증되었다.

결론적으로 임베디드 시스템(Raspberry PI 5)의 온 디바이스 AI로는 BERT-base 모델보다 DistilBERT와 같은 경량화 모델을 사용하는 것이 더 적합함을 알 수 있다. 따라서, 향후 DistilBERT 모델이 다른 임베디드 시스템에서도 온 디바이스 AI로 활용될 수 있을 것으로 예상된다.

참고문헌

[1] Victor Sanh et al., "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.", arXiv:1910.01108, 2019

[2] Geoffrey Hinton et al., "Distilling the Knowledge in a Neural Network.", arXiv:1503.02531, 2015

[3] https://alexnim.com/coding-projects-knowledge-distillation.html