소형 언어 모델 가속기를 위한 기술 분석

(Technology Analysis for Small Language Model Accelerators)

박 시 형†, 이 제 민‡, 김 병 수†, 전 석 훈†*
†한국전자기술연구원, ‡한국전자통신연구원
(Sihyeong Park, Jemin Lee, Byung-Soo Kim, Seokhun Jeon)
(†Korea Electronics Technology Institute, ‡Electronics and Telecommunications Research Institute)

Abstract : 소형 언어 모델은 엣지 장치와 같이 자원이 제한된 환경에서도 실행할 수 있는 모델이다. 이러한 소형 언어 모델은 대규모 언어 모델에 비해 적은 양의 데이터와 경량화된 모델 구조를 사용하여 크기와 복잡성을 줄였다. 소형 언어 모델을 엣지 환경에서 효율적으로 동작시키기 위해서는 최적의 연산 수행이 가능하도록 전용 가속기가 필요하다. 본 논문에서는 소형 언어 모델의 특성과 온-디바이스에서의 실행 필요성 및 소형 언어 모델 가속기를 위한 요구 사항들을 분석하였다.

Keywords : 소형 언어 모델, 엣지 장치, 소형 언어 모델 가속기, 가속기 요구 사항 분석

Ⅰ. 서 론

대규모 언어 모델 (Large Language Model, LLM)의 발전으로 인해 이러한 모델들이 다양한 분야에 적용되고 있다. Meta Llama 3.1 [1]은 최대 405B, Google Gemini [2]는 최대 540B 개의 파라미터로 구성되어 있어서 데이터 센터에서 학습이 가능하며, 추론을 위해서도 많은 하드웨어 자원이 필요하다. 따라서 이러한 LLM은 엣지 장치에서 실행이 어렵다.

소형 언어 모델 (Small Language Model, SLM)은 LLM의 구조와 복잡도를 줄여 비교적 적은 하드웨어 자원으로 실행이 가능한 모델이다. Microsoft의 Phi-3.5 [3]와 같이 LLM과 비교해 적은 3.8B 개의 파라미터를 가지는 SLM들이 제안되고 있다. 모델의 규모와 학습에 사용된 데이터 세트의 양이 늘어날수록 언어 모델의 정확도가 향상되므로 [4], SLM은 LLM보다 낮은 정확도를 가진다.

본 논문에서는 엣지 장치에서 SLM의 최적 실행을 위한 가속기를 개발하기 위해 SLM과 가속기 요구 사항에 대한 분석을 수행하였다. 특히, 엣지 환경을 위해 저전력, 고성능 및 CPU, GPU와 같은 이기종 하드웨어 지원에 대한 필요성과 이를 위한 요구 사항들을 다룬다.

*Corresponding Author(seokhun.jeon@keti.re.kr)
박시형, 김병수, 전석훈: 한국전자기술연구원
이제민: 한국전자통신연구원

※ 이 논문은 2022년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 자원을 받아 수행된 연구임 (RS-2022-II220441, 재구성형 PIM 디바이스 기반의 Memory-Centric 아키텍처 개발)

Ⅱ. 소형 언어 모델

기존의 ChatGPT [5], Llama 등의 대표적인 LLM들은 거대한 모델 크기로 인해서 많은 하드웨어 자원이 필요하다. Llama 3.2의 경우 최대 405B 개의 파라미터를 가져 추론 작업을 위해서 약 972GB (FP16)의 메모리가 필요하다. 이를 위해 NVIDIA H100와 같은 GPU가 12개 이상 필요하다. 이러한 점은 제한된 하드웨어 환경이나 엣지 장치에서 언어 모델의 실행을 어렵게 한다.

LLM의 거대한 모델 크기를 해결하기 위해서 SLM이 제안되고 있다. SLM은 모델 크기와 계산 복잡도를 줄여 엣지 장치에서의 언어 모델 실행을 가능하게 한다. SLM은 다양한 분야의 데이터로 학습된 LLM에 비해 특정 도메인이나 작업에 특화된 데이터를 사용해 학습하는 경우가 많으므로 전문적인 모델의 개발에 사용된다. 또한, 사용자와의 상호작용이 필요한 애플리케이션에서 SLM은 작은 모델 크기와 복잡성을 줄여 빠른 처리 속도를 제공할 수 있다. IoT나 모바일 장치에서는 에너지 효율이 중요하므로, SLM을 사용해 처리 시간을 단축함으로써 에너지 사용을 최소화할 수 있다.

표 1. 소형 언어 모델
Table 1. Small Language Models

모델	개발사	모델 크기 (파라미터 수)	공개 시기
Llama 3.2	Meta	1B	2024
TinyLlama	TinyLlama	1.1B	2024
phi-1.5	Microsoft	1.3B	2023
KOSMOS-2	Microsoft	1.6B	2023
Stable LM	StableLM	1.6B	2024
Gemma	Google	2B	2024
Dolly-v2	Databricks	3B	2023
Mistral	Mistral AI	7B	2023
Orca2	Microsoft	7B	2023
SOLAR	Upstage	10.7B	2023

최근에는 다양한 종류의 SLM이 연구되고 있으며 SLM의 예는 표 1과 같다. SLM은 LLM에 비해 적은 파라미터를 가져 학습, 추론에 필요한 메모리를 줄일 수 있다. 현재 연구되고 있는 많은 SLM이 인코더 혹은 디코더를 사용한 트랜스포머 구조, 특히 멀티-헤드 어텐션 (Multi-head Attention, MHA), 그룹 쿼리 어텐션 (Grouped Query Attention, GQA) 구조를 사용한다 [6]. SLM 중 TinyLlama [7], Stable LM [8]는 각각 GQA, MHA 구조로 되어 있으며 활성화 함수로 SiLU [9]를 사용한다. 대부분의 언어 모델은 SiLU와 GELU [10] 활성화 함수를 적용하고 있다. 따라서 SLM 가속기는 이러한 구조를 최적으로 실행할 수 있도록 설계되어야 한다.

phi-1.5와 같은 모델은 지도 학습 미세 조정, 근접 정책 최적화 (Proximal policy optimization, PPO) [11] 등의 방법을 사용해 모델을 최적화하여 약 1.3B의 파라미터를 가진다. Phi-1.5 모델의 추론을 위해서 2.6GB (FP16) 정도의 메모리가 필요하다. 하지만 학습을 위해서는 약 10GB 이상의 메모리가 필요하므로 많은 수의 엣지 장치에서는 수행하기 어렵다. 그러나 미세 조정으로 일부 파라미터만 튜닝하면 학습에 사용되는 메모리를 줄일 수 있다 [12].

현재 엣지 장치에서의 LLM, SLM 기반의 서비스들은 데이터 센터, 클라우드로의 오프로딩을 통해 사용자의 요청을 처리한다. 이러한 부분은 서버 사용에 따른 전력 소모, 네트워크 환경에 따른 지연 시간 증가와 사용자의 데이터를 서버로 보내므로 개인정보 노출 문제가 발생할 수 있다. 따라서 엣지 장치에서 온-디바이스로 언어 모델 처리에 대한 필요성이 증가하고 있다. 하지만 엣지 장치는 하드웨어 자원이 제한되므로 언어 모델을 효율적으로 처리할 수 있는 전용 가속기가 필요하다. 온-디바이스로 언어 모델을 처리함에 따라 개인정보 보호, 신뢰성, 빠른 응답 속도, 비용 절감, 개인화된 모델 개발 등이 가능하다.

Ⅲ. 소형 언어 모델 가속기의 요구 사항

엣지 장치에서 SLM 실행을 위해서는 저전력, 고효율의 연산이 요구된다. 이를 위해 MHA, GQA 및 GELU, SiLU 등에 최적화된 실행과 엣지 장치를 위해 특화된 부분이 필요하다. SLM 연산을 위한 가속기의 요구 사항은 다음과 같다:

- **메모리:** 엣지 장치에서 언어 모델의 학습 및 추론을 위해 기존의 엣지 장치와는 달리 비교적 큰 메모리 용량이 요구된다. 엣지 환경에 많이 사용되는 마이크로컨트롤러인 STM32와 같은 장치는 100KB 미만의 메모리를 가지지만 SLM의 실행을 위해서는 GB 단위의 메모리가 필요하다. 하지만 모바일 장치와 같은 엣지 장치는 LPDDR와 같은 메모리를 통해 8GB, 16GB 등의 비교적 큰 메모리를 탑재하는 추세로, SLM의 실행이 가능해지고 있다.
- **낮은 전력 소모:** 엣지 장치는 배터리를 사용하는 환경에서 동작할 수도 있으므로 저전력으로 언어 모델을 실행할 수 있어야 한다. 특히, LLM 실행을 위한 가속기들은 700~900W (NVIDIA H100, Intel Gaudi 3)의 높은 전력이 필요하므로 엣지 환경에서 사용할 수 없다. 엣지 인공지능 환경에 널리 적용되는 NVIDIA Jetson의 경우 약 20W의 열 설계 전력 (Thermal Design Power, TDP)을 가진다. 낮은 전력 소모를 위해서는 연산 최적화와 가속기-메모리 간의 데이터 전송 최적화 등의 고려가 필요하다.
- **이기종 하드웨어 지원:** 최근 NVIDIA Jetson과 모바일 장치 등의 엣지 환경에서는 CPU 이외에도 GPU, 딥 러닝 가속 하드웨어가 함께 탑재되고 있다. SLM 연산의 효율적인 실행을 위해서는 가속기에서만 실행하는 것이 아니라, 연산의 종류에 따라 여러 하드웨어에서 병렬적으로 실행하는 것이 최적의 결과를 보일 수 있다. 이를 위해서는 SLM의 병렬

실행을 위한 방법 [13]을 적용할 수 있으며, 각 하드웨어에서 최적으로 실행할 수 있는 연산이 분석되어야 한다. 또한, 메모리에서 각 하드웨어로 데이터를 전송하는 시간과, 각 하드웨어가 동기화를 위한 시간 등을 고려해야 최적의 연산 실행이 가능하다. 특히, 각 SLM 가속기의 하드웨어 구조에 적합하도록 입력 데이터의 타일링, 하드웨어 간의 파이프라이닝 등이 고려되어야 한다 [14].

Ⅳ. 결 론

본 논문에서는 엣지 장치에서 SLM의 효율적인 실행을 위한 가속기 기술을 분석하였다. LLM의 거대한 크기와 높은 자원 요구 사항은 엣지 환경에서 적용을 제한하며, 이를 극복하기 위해 SLM의 필요성이 대두되고 있다. SLM은 모델의 크기와 복잡도를 줄여 제한된 하드웨어 자원에서도 동작이 가능하지만, 최적의 성능을 위해서는 전용 가속기의 지원이 필요하다.

본 논문에서는 SLM의 구조적 특징과 엣지 환경의 요구 사항을 고려하여 가속기의 핵심 요구 사항을 도출하였다. 구체적으로, 메모리 용량의 효율적인 활용, 낮은 전력 소모, 이기종 하드웨어 지원 등이 중요함을 보였다.

References

[1] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan, et al., “The llama 3 herd of models”. arXiv preprint arXiv:2407.21783, pp. 1-92, 2024.

[2] R. Anil, M. Faruqui, S. Borgeaud, J.B. Alayrac, J. Yu, et al., “Gemini: a family of highly capable multimodal models”. arXiv preprint arXiv:2312.11805, pp. 1-90, 2023.

[3] M. Abdin, X. Jin, A. Salim, J. Aneja, N. Karampatziakis, et al., “Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone”. arXiv preprint arXiv:2404.14219, pp. 1-24, 2024.

[4] Tom.B. Brown, B. Mann, N. Ryder, M. Subbiah, et al., “Language Models are Few-Shot Learners”. arXiv preprint arXiv:2005.14165, pp. 1-75, 2020.

[5] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F.L. Aleman, D Almeida, J Altenschmidt, S. Altman, S. Anadkat, et al., “Gpt-4 technical report”. arXiv preprint arXiv:2303.08774, pp.1-100, 2023.

[6] Z. Lu, X. Li, D. Cai, R. Yi, F. Liu, X. Zhang, N.D. Lane, and M. Xu, “Small Language Models: Survey, Measurements, and Insights”. arXiv preprint arXiv:2409.15790, pp. 1-22, 2024.

[7] P. Zhang, G. Zeng, T. Wang, and W. Lu, “TinyLlama: An Open-Source Small Language Model”. arXiv preprint arXiv:2401.02385, pp. 1.-10, 2024.

[8] M. Bellagente, J. Tow, D. Mahan, D. Phung, M. Zhuravinskyi, et al., “Stable LM 2 1.6B Technical Report”. arXiv preprint arXiv:2402.17834, pp. 1-23, 2024.

[9] S. Elfwing, E. Uchibe, and K. Doya, “Sigmoid-weighted linear units for neural network function approximation in reinforcement learning”. Neural Networks, Vol. 107, pp. 3-11, 2018.

[10] D. Hendrycks, and K. Gimpel, “Gaussian Error Linear Units (GELUs)”. arXiv preprint arXiv:1606.08415, pp. 1-10, 2016.

[11] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, Proximal policy optimization algorithms”. arXiv preprint arXiv:1707.06347. pp. 1-12, 2017.

[12] Y. Zhang, P. Li, J. Hong, J. Li, Y. Zhang, W. Zheng, et al., “Revisiting zeroth-order optimization for memory-efficient llm fine-tuning: A benchmark”. arXiv preprint arXiv:2402.11592. pp. 1-18, 2024.

[13] F. Brakel, U. Odyurt, and A.L. Varbanescu, “Model Parallelism on Distributed Infrastructure: A Literature Review from Theory to LLM Case-Studies”. arXiv preprint arXiv:2403.03699, pp. 1-10, 2024.

[14] H. Xia, Z. Zheng, Y. Li, D. Zhuang, Z. Zhou, X. Qui, Y. Li, W. Lin, and S.L,. Song, “Flash-llm: Enabling cost-effective and highly-efficient large generative model inference with unstructured sparsity”. arXiv preprint arXiv:2309.10285, pp. 1-14, 2023.