한국정보기술학회논문지

The Journal of Korean Institute of Information Technology

Journal of KIIT. Vol. 23, No. 6, Jun. 30, 2025. pISSN 1598-8619, eISSN 2093-7571

한국연구재단 등재지

This journal was supported by the Korean Federation of Science and Technology Societies(KOFST) grant funded by the Korean government.

사단법인 한국정보기술학회
Korean Institute of Information Technology
http://www.ki-it.or.kr/ http://ki-it.com/

The Journal of Korean Institute of Information Technology

Journal of KIIT. Vol. 23, No. 6, Jun. 30, 2025. pISSN 1598-8619, eISSN 2093-7571

Korean Institute of Information Technology
http://www.ki-it.or.kr or http://www.ki-it.com

Journal of KIIT. Vol. 23, No. 6, pp. 75-89, Jun. 30, 2025. pISSN 1598-8619, eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2025.23.6.75

대규모 언어 모델 가속을 위한 상용 가속기 기술 동향

박시형*[1], 이제민**, 김병수*[2], 전석훈*[3]

A Survey of Proprietary Accelerators for Large Language Models

Sihyeong Park*[1], Jemin Lee**, Byung-Soo Kim*[2], and Seokhun Jeon*[3]

이 논문은 2025년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임
(No.RS-2024-00402898, 시뮬레이션 기반 고속/고정확도 데이터센터 워크로드/시스템 분석 플랫폼 개발)

요 약

대형 언어 모델(LLMsLarge Language Models)의 규모가 빠르게 확대되고, 동시에 모델 학습 및 추론 비용이 기하급수적으로 증가함에 따라 이를 효율적으로 처리하기 위한 전용 하드웨어 가속기의 중요성이 커지고 있다. 본 논문에서는 LLM 가속기의 필요성을 논의하고, NVIDIA, AMD, Google 등 주요 기업에서 개발한 상용 Application-Specific Integrated Circuit 기반 LLM 가속기의 하드웨어 및 소프트웨어 특성을 메모리 아키텍처, 성능, 인터커넥션, 소프트웨어 스택 등 다양한 관점에서 종합적으로 분석한다. 또한 LLM 가속기가 직면한 메모리 용량, 전력 효율, 확장성, 소프트웨어 호환성 등의 주요 기술적 도전 과제를 설명하고, 하드웨어 개발 방안을 검토한다. 이러한 분석을 바탕으로 혼합 정밀도 지원, 대규모 분산 학습을 위한 고속 인터커넥션, 컴파일러 등의 핵심 고려 사항을 제시하고, 나아가 차세대 LLM 가속기 개발에 필요한 연구 방향을 제안한다.

Abstract

As large language models (LLMs) continue to expand in scale, their training and inference costs grow exponentially, underscoring the importance of specialized hardware accelerators for efficient processing. This paper examines the necessity of such accelerators through a comprehensive analysis of major proprietary Application-Specific Integrated Circuit solutions, covering memory architecture, performance, interconnects and software stacks. In addition we identify key challenges such as memory capacity, power efficiency, scalability and software framework of LLM accelerators. Based on these insights we outline critical considerations for next-generation LLM accelerators including mixed-precision support, high-speed interconnects for large-scale distributed training and compiler optimizations.

Keywords
large language model, proprietary accelerator, ASIC, performance, software

* 한국전자기술연구원 SoC플랫폼 연구센터(*[3] 교신저자)
- ORCID[1]: https://orcid.org/0000-0001-8244-4817
- ORCID[2]: https://orcid.org/0009-0004-2996-2841
- ORCID[3]: https://orcid.org/0009-0002-4009-178X
** 과학기술연합대학원대학교 인공지능학과 조교수
- ORCID: https://orcid.org/0000-0002-9332-3508

· Received: Mar. 19, 2025, Revised: May 08, 2025, Accepted: May 11, 2025
· Corresponding Author: Seokhun Jeon
SoC Platform Research Center, Korea Electronics Technology Institute
43, Changeop-ro, Sujeong-gu, Seongnam-si, Gyeonggi-do, Korea
Tel.: +82-31-785-6718, Email: seokhun.jeon@keti.re.kr

Ⅰ. 서 론

ChatGPT[1]의 성공 이후, 다양한 대형 언어 모델(LLM, Large Language Model)에 관한 연구와 이를 활용한 서비스들이 개발되고 있다. 최근에는 단순한 챗봇과 같은 텍스트 생성 모델 이외에도 텍스트, 이미지, 오디오, 비디오 등을 처리하는 멀티모달 LLM[2]이 제안되고 있으며, 이러한 LLM 모델들의 구조적 복잡성 증가에 따라 모델 학습에 사용되는 데이터 규모도 가파르게 확대되고 있다. 자연스럽게 학습과 추론에 필요한 하드웨어 자원과 비용이 기하급수적으로 상승함에 따라, LLM 연산에 특화된 전용 하드웨어를 활용해 전력 및 냉각 비용을 줄이려는 시도가 늘고 있다[3].

특히 DeepSeek-R1[4]과 같은 추론(Reasoning) 모델뿐만 아니라 AI(Artificial Intelligence) 에이전트(Agent)[5]의 발전으로 추론 과정이 더 복잡해지고, 길어짐에 따라 효율적인 처리에 대한 필요성이 증가하고 있다.

일반적으로 LLM의 학습과 추론은 병렬 연산 성능이 뛰어난 그래픽 처리 장치(GPU, Graphics Processing Unit)에 의존하며, GPU는 범용성과 풍부한 소프트웨어 스택 지원을 강점으로 가진다. 그러나 GPU는 높은 전력 소모로 인한 발열 문제와 이를 해결하기 위한 냉각 시스템 운영 비용이 막대하다는 단점이 존재한다. 따라서 대형 언어 모델 연산에 최적화된 전용 하드웨어, 즉 LLM 가속기(Accelerator)를 활용하여 연산 비용과 에너지 소비를 절감하려는 노력이 필요하다.

LLM 가속기는 LLM 연산을 가속하기 위해 특화된 하드웨어로, GPU와 달리 전용 회로(ASIC, Application-specific Integrated Circuit)를 통해 더 높은 에너지 효율과 연산 최적화를 제공할 수 있다. 다만, 제조사별 하드웨어 및 소프트웨어 스택이 달라 호환성 문제가 발생할 가능성이 있고, vLLM[6], DeepSpeed[7] 등의 LLM 프레임워크 지원과, 하드웨어별 최적화 코드를 생성하는 컴파일러 기술이 점차 중요해지고 있다.

기존 LLM 연구들은 모델 최적화[8], 효율적인 모델 구조[9], 추론 최적화[10] 등 주로 모델 중심의 분석에 초점을 맞춰 진행되었다. 특히 대부분의 연구는 NVIDIA GPU를 기반으로 수행되었으며, LLM 학습 및 추론 환경이 GPU 아키텍처에 최적화되어 왔다. 그러나 최근 다양한 LLM 전용 가속 하드웨어가 개발됨에 따라, 하드웨어별 특성과 적용 가능성을 다각도로 분석하는 연구가 필요하다.

LLM 추론 최적화를 위한 하드웨어 기술들이 연구되었지만, 상용화 단계보다는 연구적인 수준이거나, 배열 연산의 희소(Sparsity) 최적화나 양자화(Quantization)를 적용해 연산 일부를 효율적으로 수행하는 데에 집중하고 있다[11][12]. 따라서 실제 LLM 응용 실행에서 제한점을 가진다.

본 논문에서는 대형 언어 모델 연산의 최적 수행을 위한 가속기의 개념과 요구사항을 살펴보고, 상용(COTS, Commercial Off-The-Shelf) LLM 가속기들의 하드웨어 및 소프트웨어 특성을 분석한다. 특히 ASIC 기반으로 상용화된 제품을 중심으로, 공개된 연구 논문, 백서(White paper) 등을 통해 접근할 수 있는 가속기를 검토한다. 본 논문의 목적은 이러한 LLM 가속기의 주요 특징과 과제를 체계적으로 정리하고, 향후 연구 방향을 제시하는 데 있다.

논문의 구성은 다음과 같다. 2장에서는 LLM 가속기의 개념을 설명하고, 3장에서는 주요 상용 LLM 가속기를 하드웨어와 소프트웨어 측면에서 분석한다. 4장에서는 앞선 내용을 바탕으로 LLM 가속기가 직면한 도전 과제를 논의하며, 5장에서 결론을 제시한다.

Ⅱ. 대규모 언어 모델 가속기

LLM 가속기는 대형 언어 모델의 학습과 추론을 가속화하고 연산 효율을 극대화하도록 설계된 특수 하드웨어이다. 최근 LLM의 모델 크기와 연산 복잡도가 급증하면서, 기존 범용 하드웨어(GPU 등)는 성능 및 전력 효율에서 한계를 드러내고 있다. 이에 따라 고성능 전용 하드웨어를 도입해 자연어 처리, 음성 인식, 코드 생성, 데이터 분석 및 자동화 등 다양한 분야에서 요구되는 실시간 응답성과 비용 효율성을 개선하려는 노력이 활발하다.

LLM 가속기는 텐서 연산(Tensor operations) 최적

화와 메모리 계층을 효과적으로 설계하여 에너지 효율을 높이고 처리 속도를 극대화함으로써 대기 시간(Latency)을 단축한다. 이는 대규모 AI 서비스를 제공하는 클라우드 환경뿐 아니라, 엣지 컴퓨팅(Edge computing) 환경에서도 중요한 요소로 작용하고 있다. 본 장에서는 LLM 가속기의 하드웨의 구조와 특성에 대해 설명한다.

2.1 가속기 구현 방식

LLM 가속기는 대표적으로 GPU, FPGA (Field-Programmable Gate Array), ASIC 기반으로 구현될 수 있으며, 각 방식은 연산 구조, 확장성, 전력 효율 등의 측면에서 다른 특성을 보여 특정 워크로드(Workload)에 따라 적절한 선택이 필요하다.

GPU. GPU는 병렬 연산 성능이 뛰어나며, 범용 컴퓨팅을 지원하는 소프트웨어 스택이 잘 갖추어져 있어 LLM 학습과 추론에서 가장 널리 사용되는 가속기 형태이다. GPU는 수천 개 이상의 연산 코어를 보유하고 있으며, 행렬 연산(Matrix computation) 및 텐서 연산을 효율적으로 수행하는 텐서 코어(Tensor core)와 같은 전용 연산 유닛을 포함하고 있다. 특히, NVIDIA CUDA 및 AMC ROCm(Radeon Open Compute)와 같은 프로그래밍 프레임워크/라이브러리를 제공하여 기존 머신 러닝 및 딥러닝 연구에서 가장 널리 채택되고 있다.

GPU는 높은 연산 성능을 제공하지만, 전력 소모가 크고 메모리 대역폭이 병목이 될 가능성이 높다는 단점이 있다. 또한, LLM의 규모가 커짐에 따라 이를 처리하기 위해 다수의 GPU를 병렬로 연결하는 NVLink, 인피니티 패브릭(Infinity fabric)과 같은 인터커넥션(Interconnection) 기술이 필수적으로 자리 잡고 있다. 최근에는 GPU에서 LLM 학습 및 추론을 위해 혼합 정밀도(FP16, INT8, INT4 등) 연산을 지원하여 연산 속도를 높이고 메모리 사용량을 줄이는 기법이 널리 활용되고 있다[13][14].

FPGA. FPGA 기반 LLM 가속기는 프로그래밍할 수 있는 논리 소자를 활용하여 하드웨어 수준에서 연산 로직을 동적으로 변경할 수 있는 가속기이다. LLM 연산을 포함한 특정 워크로드에 맞게 사용자가 직접 연산 회로를 설계하고 최적화할 수 있으며, 기존 GPU 대비 높은 연산 효율을 제공할 수 있다. 특히, 고속 데이터 스트리밍(Data streaming) 및 데이터 흐름(Dataflow) 처리에 유리하여 특정 연산에 대한 성능 최적화를 수행할 수 있다. 따라서 FPGA 기반 LLM 가속기도 활발히 연구되고 있다[15][16].

FPGA의 장점은 유연한 아키텍처 구조와 낮은 전력 소모이다. ASIC처럼 특정 연산에 최적화할 수 있으면서도, 필요에 따라 하드웨어 구성을 변경할 수 있으므로 신규 LLM 모델 구조 및 연산 방식 변화에 빠르게 대응할 수 있다. 그러나 모델에 최적화된 회로 구성을 위한 설계와 최적화 기술에 대한 개발 시간이 길다는 단점이 있다.

ASIC. ASIC 기반 LLM 가속기는 특정 연산을 최적으로 수행하도록 맞춤 설계된 반도체 칩으로, LLM 학습 및 추론에 최적화된 하드웨어를 제공할 수 있다. ASIC 기반 LLM 가속기의 대표적인 예로는 Google의 TPU(Tensor Processing Unit)[17], Cerebras WSE-2[18], SambaNova RDU[19] 등이 있으며, 이러한 칩들은 LLM 연산의 주요 병목을 고려하여 메모리 대역폭, 연산 속도, 전력 소모를 최적화하는 것을 목표로 한다.

ASIC의 가장 큰 장점은 고정된 연산 구조를 기반으로 GPU 및 FPGA 대비 높은 에너지 효율과 처리 속도를 제공할 수 있다는 점이다. 특히, HBM (High Bandwidth Memory)과 같은 고속 메모리를 활용하여 대규모 LLM 연산에 최적화된 데이터 흐름을 구성할 수 있다. 그러나 설계 및 제조 비용이 매우 높고 한 번 제작된 ASIC은 하드웨어 구조를 변경할 수 없어 새로운 LLM 모델에 대한 유연성이 떨어질 수 있다는 단점이 있다. 따라서 대규모 데이터센터에서 장기적인 운영 비용 절감과 성능 최적화를 목적으로 활용되는 경우가 많다.

이처럼 GPU, FPGA, ASIC은 각각의 특성과 장단점이 존재하며, LLM의 학습과 추론 요구사항에 따라 적절한 하드웨어를 선택하는 것이 중요하다. 최근에는 CPU와 FPGA, ASIC을 혼합해서 LLM 연산을 최적화하는 방법도 연구되고 있다[12].

일반적으로 상용 LLM 가속기들은 ASIC 기반의 형태를 가지므로 본 논문에서는 ASIC 기반 LLM 가속기에 집중한다.

2.2 가속기 탑재 메모리

LLM 가속기들은 대규모 파라미터를 처리하는 LLM의 높은 메모리 사용량과 메모리 바운드(Memory-bound) 작업을 효과적으로 지원하기 위해 대체로 HBM을 탑재하고 있다[20]. HBM은 기존 DRAM(Dynamic Random Access Memory) 기반 메모리보다 높은 대역폭을 제공하여, 대규모 행렬 연산 및 어텐션(Attention) 연산과 같은 LLM 연산의 병목 현상을 줄이는 데 유리하다.

그러나 일부 LLM 가속기는 SRAM(Static Random Access Memory), LPDDR(Low Power DDR) 등의 메모리를 활용하고 있다. 이러한 메모리는 각각의 장단점이 존재하며, 가속기 설계 및 사용 목적에 따라 달라질 수 있다.

HBM. HBM의 경우 높은 메모리 대역폭을 통해 메모리 작업에 대해 낮은 대기 시간 제공과 대규모 데이터 스트리밍 최적화가 가능하다. 하지만 2.5D, 3D 스택 패키징 기술을 사용하므로 제조 비용이 많이 들고 발열 문제가 발생할 수 있다[21].

SRAM. SRAM의 경우 DRAM 대비 낮은 대기 시간과 빠른 접근 속도, 높은 전력 효율을 제공한다. 하지만 메모리 밀도가 낮아 용량이 작고, 비용이 높아 LLM의 실행을 위해 다량의 SRAM이 필요해 칩 면적과 제조 비용이 증가할 수 있다.

LPDDR. LPDDR은 기존 DDR 메모리 대비 저전력으로 동작하며 비용이 비교적 낮고, 용량 확장이 쉬운 장점을 가져 모바일, 엣지 환경에 탑재된다. 하지만 HBM 대비 메모리 대역폭이 낮아 LLM 연산에서 메모리 병목이 발생할 수 있다.

III. 대규모 언어 모델 가속기 기술 동향

LLM의 학습과 추론 과정은 매우 높은 연산량과 대규모 메모리 요구 사항을 수반하며, 이를 효과적으로 처리하기 위해 다양한 상용 LLM 가속기가 개발되고 있다.

본 장에서는 상용 LLM 가속기의 하드웨어 및 소프트웨어 특성을 분석하고, 각 가속기의 연산 성능, 메모리 아키텍처, 전력 소모, 확장성, 인터커넥션 기술, 소프트웨어 스택 등의 요소를 비교 및 검토한다. 이를 위해 상용 ASIC 기반의 LLM 가속기 NVIDIA H100[22], AMD MI300X[23], Cerebras WSE-2[18], Google TPU v4[17], Graphcore IPU[24], Groq LPU[25], Intel Gaudi 3[26], SambaNova SN40L[19], Tenstorrent Grayskull[27]의 주요 특징을 살펴본다. 표 1은 각 LLM 가속기의 하드웨어 스펙을 나타낸다. 표의 최신 릴리즈는 해당 가속기의 현재 최신 하드웨어 버전을 의미한다.

표 1. 대규모 언어 모델 가속기 하드웨어 스펙
Table 1. LLMl accelerator hardware specifications

Accelerator	Technology node	TDP (W)	Memory			Release date	Latest release	Supported operation	
			Type	Capacity	Bandwidth			Training	Inference
NVIDIA H100	4nm	700	HBM3	80GB	2.25TB/s	2022	GB200	O	O
AMD MI300X	6nm/5nm	750	HBM3	192GB	5.2TB/s	2023	-	O	O
Cerebras WSE-2	7nm	20k	SRAM	40GB	20PB/s	2012	WSE-3	O	O
Google TPU v4	7nm	192	HBM2	32GB	1.2TB/s	2020	TPU v6 (Trillium)	O	O
Graphcore IPU	7nm	300	SRAM	0.9GB	65TB/s	2020	-	O	O
Groq LPU	14nm	300	SRAM	230MB	80TB/s	2024	-	X	O
Intel Gaudi3	5nm	900	HBM2E	128GB	3.7TB/s	2024	-	O	O
SambaNova SN40L	5nm	-	HBM3, DDR5	64GB, 1.5TB	2TB/s, 200GB/s	2023	-	O	O
Tenstorrent Grayskull	12nm	200	LPDDR4	8BB	118.4GB/s	2024	Wormhole	X	O

3.1 하드웨어 특성

3.1.1 하드웨어 구성

NVIDIA H100[22]은 LLM 학습과 추론에 널리 사용되는 가속기로, NVIDIA Hopper 아키텍처를 기반으로 한다. H100은 132개의 스트리밍 멀티프로세서(SM, Streaming Multiprocessor)로 구성되어 있으며 각 SM에는 텐서 코어와 캐시(Cache) 등이 포함된다. H100은 80GB의 HBM3를 탑재해 3TB/s 이상의 메모리 대역폭을 제공한다.

AMD MI300X[23]는 칩렛(Chiplet) 구조를 기반으로 가속을 위한 8개의 엑셀러레이터 컴플렉스 다이(XCD, Accelerator Complex Die)에 각각 2개의 HBM3를 연결함으로써, 총 192GB 메모리와 5TB/s 이상의 메모리 대역폭을 지원한다.

Cerebras WSE-2[18]는 웨이퍼 스케일 칩(Wafer-scale chip)으로, 일반적인 GPU 칩(NVIDIA A100, 826mm²) 대비 면적이 46,255mm²로 약 55배 이상 크며, 85만 개의 코어와 40GB의 SRAM을 갖추고 있다. 그러나 20kW 이상의 전력을 소비한다는 단점이 있다.

Google TPU v4[17]는 칩당 2개의 텐서코어(TensorCore)로 구성되어 있으며, 칩당 32GB의 HBM2를 탑재해 1.2TB/s의 대역폭을 제공한다. 광학적으로 재구성할 수 있는 네트워크 구조를 채택해 최대 4,096개의 칩을 연결할 수 있고, 이를 통해 슈퍼컴퓨터 규모의 대규모 분산 처리를 수행할 수 있다.

Graphcore IPU[24](Colossus MK2 IPU 프로세서)는 IPU마다 1,472개의 독립적인 프로세서 코어가 있어 대규모 병렬 처리에 특화되어 있다. 각 IPU는 900MB의 SRAM을 탑재하며, 타일(Tile) 단위로 프로세싱 유닛과 SRAM을 배치해 인프로세서 메모리(In-Processor memory) 구조 기반의 빠른 통신을 제공한다.

Groq LPU[25]는 소프트웨어 정의 텐서 스트리밍 프로세싱(Software-Defined TSP(Tensor Streaming Processing)) 아키텍처를 활용해 대규모 연산을 수행한다. 최대 10,440개의 TSP를 네트워크 인터커넥트로 확장 가능하며 230MB의 SRAM을 탑재한다. 하지만 예를 들어 Llama 70B 모델[25]을 구동하려면 작은 메모리 용량으로 인해 수백 개의 LPU가 필요해 전력 소모와 운영 비용 문제가 발생할 수 있다.

Intel Gaudi 3[26]는 8개의 매트릭스 곱셈 엔진(MME, Matrix Multiplication Engine)과 64개의 텐서 프로세서 코어(TPC, Tensor Processor Core)를 탑재해 높은 병렬 처리 성능을 제공한다. 128GB의 HBM2E를 갖추어 3TB/s 이상의 메모리 대역폭도 지원한다.

SambaNova SN40L[19](RDU, Reconfigura ble Dataflow Unit)은 RDU 소켓 하나에 1,040개의 분산 패턴 계산 유닛(Distributed PCU(Pattern Compute Unit))을 배치해 연산을 처리한다. 독특하게 64GB HBM, 1.5TB DRAM, 520MB SRAM의 3단계 메모리 시스템을 채택해 수백 개의 복잡한 연산을 단일 커널로 융합하는 스트리밍 데이터플로 병렬화를 지원한다.

Tenstorrent Grayskull[27]은 RISC-V 기반 아키텍처를 사용하고, 칩 내 네트워크(NoC, Network-on-Chip) 구조를 통해 최대 120개의 텐식스 코어(Tensix core)를 연결함으로써 지능형 데이터 이동과 처리를 수행한다. 8GB의 LPDDR4 메모리를 탑재해 118GB/s 이상의 대역폭을 제공하나, LLM 추론 시 코어 수 대비 메모리가 부족해 많은 장치를 병렬로 연결해야 할 수 있으며, 이에 따라 높은 운영 비용이 발생할 수 있다.

3.1.2 하드웨어 성능

각 LLM 가속기의 성능은 표 2에 정리되어 있으며, 지원하는 데이터 타입과 연산 방식에 따라 성능 차이가 발생한다. 그림 1은 가속기별 열 설계 전력(TDP, Thermal Design Power) 대비 FP16 성능이다.

NVIDIA H100[22]과 AMD MI300X[23]는 LLM의 학습과 추론을 모두 지원하며, 다양한 데이터 타입(Data Type)을 처리할 수 있도록 설계되었다. 특히, FP32, TF32, FP16, BF16, INT8, INT4 등 혼합 정밀도(Mixed precision) 연산을 지원하여 학습 및 추론 시 성능과 메모리 효율성을 최적화할 수 있다.

표 2. 대규모 언어 모델 가속기 성능
Table 2. Performance of large language model accelerator

Accelerator	Data Type					
	FP8	FP16	BF16	INT8	FP32	FP64
NVIDIA H100	3,958 TFLOPS	1,979 TFLOPS	1,979 TFLOPS	3,958 TOPS	67 TFLOPS	34 TFLOPS
AMD MI300X	20.9 PFLOPS	10.5 PFLOPS	10.5 PFLOPS	20.9 POPS	1.3 PFLOPS	1.3 PFLOPS
Cerebras WSE-2	-	75 PFLOPS	-	-	-	-
Google TPU v4	-	-	275 TFLOPS	275 TOPS	-	-
Graphcore IPU	-	250 TFLOPS	-	-	62 TFLOPS	-
Groq LPU	-	205 TFLOPS	-	820 TOPS	-	-
Intel Gaudi3	1,835 TFLOPS	459 TFLOPS	1,835 TFLOPS	-	229 TFLOPS	-
SambaNova N40L	-	-	638 TFLOPS	-	-	-
Tenstorrent Grayskull	332 TFLOPS	92 TFLOPS	-	-	-	-

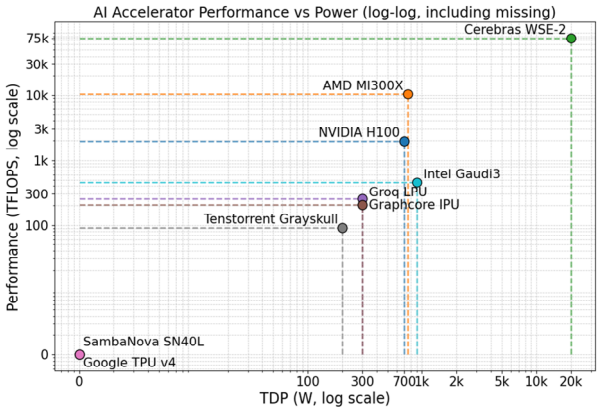


그림 1. 열 설계 효율과 가속기의 FP16 성능
Fig. 1. FP16 performance of accelerators and thermal design power

Google TPU v4[17], Cerebras WSE-2[18], Intel Gaudi 3[26], Graphcore IPU[24], SambaNova SN40L[19]는 LLM의 학습 및 추론을 지원하지만, 특정 연산 효율성을 높이기 위해 일부 데이터 타입만 최적화하여 지원한다.

반면, Groq LPU[25]와 Tenstorrent Grayskull[27]은 추론 전용 가속기로 설계되었기 때문에 모델 학습을 지원하지 않으며, 한정된 데이터 타입만 최적화하여 제공한다.

각 LLM 가속기의 데이터 타입별 연산 성능은 이론상 최대 수치를 기반으로 산정된 것이므로, 실제 모델 구성과 워크로드에 따라 실제 성능 효율은 달라질 수 있다. 특히, 메모리 대역폭 및 인터커넥션 성능, LLM 모델의 크기 및 아키텍처 최적화 여부 및 연산 단위별 배치(Batch size), 시퀀스 길이(Sequence length) 등이 LLM 실행 성능에 영향을 미칠 수 있다. 따라서 하드웨어 간 단순한 성능 수치(FLOPS, FLoating point Operations Per Second) 비교만으로 성능을 평가하기는 어려우며, 실제 LLM 실행 성능을 평가할 때는 벤치마크 결과와 실사용 시나리오를 고려한 분석이 필요하다.

3.1.3 인터커넥션 구조

LLM 가속기의 또 다른 핵심 요소는 인터커넥션 기술이다. LLM 규모가 커질수록 단일 가속기로 처리하기 어려워지므로, 그림 2와 같이 가속기 내부(칩-투-칩)와 가속기 간(카드-투-카드), 노드 간(노드-투-노드) 고속 통신을 가능케 하는 인터커넥션 설계가 매우 중요해진다. 일반적으로 하나의 노드는 8개의 가속기로 구성된다. 표 3의 각 LLM 가속기의 인터커넥션 기술은 높은 대역폭 확보를 위해 이더넷이나 전용 인터페이스를 활용하며, 칩렛 구조의 발전과 함께 인터포저(Interposer) 브리지를 통해 칩투 칩 통신을 지원할 수도 있다.

NVIDIA H100[22]. NVIDIA H100은 NVLink, NVSwitch, 인피니밴드(Mellanox/NVIDIA infiniband) 등을 통해 인터커넥션을 제공한다. NVLink는 GPU 간 통신용으로 PCIe보다 높은 대역폭을 지원하며, NVSwitch는 대규모 GPU를 상호 연결하는 스위치로 네트워크 트래픽 관리에 활용된다.

표 3. 대규모 언어 모델 가속기 인터커넥션
Table 3. Interconnections of LLM accelerator

Accelerator	Chip-to-chip		Card-to-card (Node)		Node-to-node	
	Type	Bandwidth	Type	Bandwidth	Type	Bandwidth
NVIDIA H100	NVLink	900GB/s	NVLink/NVSwitch	900GB/s	InfiniBand	400Gb/s
AMD MI300X	Interposer bridge	-	Infinity Fabric	128GB/s	Infinity Fabric/PCIe	-
Cerebras WSE-2	SwarmX fabric	200Pb/s	Custom interconnection	-	-	-
Google TPU v4	Inter-Core Interconnect Link	-	Optical Circuit Switches (OCS)	50Gb/s	OCSes	50Gb/s
Graphcore IPU	IPU-Links	320Gb/s	GW-Link	100Gb/s	Sync-Link	100Gb/s
Groq LPU	Chip-to-chip (C2C) link	30Gb/s	C2C Links (Ehternet/QSFP)	-	C2C Links (Ehternet/QSFP)	-
Intel Gaudi3	Interposer bridge	-	PoCE (PAM 4 SerDes)	112Gb/s	RoCE (PAM 8 SerDes)	800Gb/s
SambaNova SN40L	Die-to-die direct connect	-	Peer-to-peer interface	-	Top Level Network	-
Tenstorrent Grayskull	NoC (Torus)	192GB/s	PCIe	-	-	-

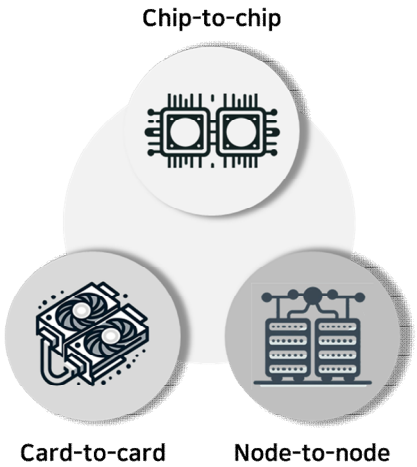


그림 2. 가속기 인터커넥션 방법
Fig. 2. Interconnection methods for accelerators

인피니밴드는 고성능 컴퓨팅(HPC, High Performance Computing) 및 데이터센터 환경에서 원격 직접 메모리 액세스(RDMA, Remote Direct Memory Access) 기능을 지원하는 고속 통신 기술이다. NVIDIA는 NCCL(NVIDIA Collective Communications Library)를 통해 여러 GPU 간 최적 병렬 통신을 지원한다.

AMD MI300X[23]는 인피니티 패브릭을 사용하여 칩 간 고대역폭 링크(양방향 최대 128GB/s)를 제공한다. NVIDIA의 NCCL과 유사하게 ROCm 통신 집합 라이브러리 RCCL(ROCm Communication Collectives Library)를 통해 다수의 LLM 가속기 간 효율적인 통신을 지원한다.

Google TPU v4[17]는 칩 간 인터컨넥트(ICI, Inter-Chip Interconnect)를 통해 칩 간 저지연 데이터 전송을 구현한다. 다수의 TPU를 연결한 TPU 팟(Pod)을 운용하기 위해 광학 회로 스위치(OCS, Optical Circuit Switch) 기반 대규모 네트워크 토폴로지(Topology)를 동적으로 재구성할 수 있어, 전자 스위치 대비 더 낮은 대기 시간과 높은 전송 용량을 지원한다.

Intel Gaudi 3[26]는 RoCE(RDMA over Converged Ethernet) 기반의 고속 통신을 지원한다. 이는 이더넷을 기반으로 RDMA 프로토콜을 구현해, 가속기 간 직접 데이터 전송을 가능하게 함으로써 저지연 환경을 마련한다. Intel은 oneCCL(oneAPI Collective Communications Library)를 통해 대규모 장치 간 통신 최적화를 지원한다.

Tenstorrent Grayskull[27]은 2D 양방향 토러스(2D

Bi-directional torus) 네트워크를 사용해 칩 내부 코어 간 데이터 교환을 수행하며, 데이터플로우 아키텍처를 바탕으로 병렬 컴퓨팅에 높은 효율을 제공한다. 카드 간 통신은 PCIe를 이용한다.

이 밖에 Graphcore IPU[24]는 IPU-Link, GW-Link 등을 활용해 가속기를 클러스터링하며, Cerebras WSE-2[18], Groq LPU[25], SambaNova SN40L[19]도 자체적으로 설계된 커스텀 인터커넥션을 통해 다수의 가속기 간 통신을 지원한다.

3.1.4 하드웨어 특성에 따른 가속기 선택

주요 LLM 가속기들은 각각의 구조적 설계 철학, 연산 최적화 방식, 그리고 메모리 계층 구조를 기반으로 성능을 극대화하는 전략을 취하며, 이를 통해 특정 LLM 워크로드에 적합한 차별화된 특성을 갖는다. 일부 가속기는 대규모 모델 학습에 최적화된 강력한 병렬 연산 성능과 HBM 기반의 높은 메모리 대역폭을 제공하는 반면, 다른 가속기들은 저전력 환경에서의 효율적인 추론 성능이나 데이터 이동 최소화를 위한 독자적인 메모리 아키텍처를 채택하는 등의 설계 방향을 따른다. 이러한 차이로 인해 가속기별로 성능, 확장성, 전력 효율 등에서 고유한 장단점이 존재하며, LLM의 학습과 추론 워크로드에 따라 최적의 가속기 선택이 필요하다.

예를 들어, FP16을 사용해서 추론하는 서비스의 경우 그림 3과 같이 AMD Instinct MI300X[23]가 가장 좋은 전력 소모 대비 성능 효율성을 가질 수 있다. 만약, 작은 배치 크기를 가지는 서비스에서는 메모리 병목이 발생할 수 있으므로 Cerebras WSE-2[18], Groq LPU[25]와 같이 메모리 대역폭이 큰 하드웨어가 적합하다. LLM의 규모가 크거나, 큰 배치 크기를 사용하는 서비스에서는 AMD MI300X[23]와 같이 메모리가 큰 시스템이 유리하다. 더불어, 다중 서버를 사용할 때는 가속기 시스템의 노드와 노드 통신 지원과 성능이 매우 중요하다. 노드 간 데이터 전송 기술이 지원되지 않으면, 네트워크로 데이터를 전송하고 동기화해야 하므로 큰 오버헤드가 발생할 수 있다. 이러한 부분을 고려해서 종합적으로 서비스에 적합한 가속기 시스템 구축이 필요하다.

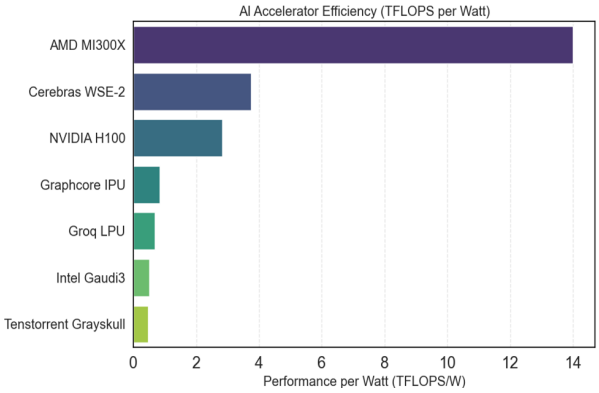


그림 3. 전력 소모 대비 FP16 성능 효율성
Fig. 3. FP16 performance efficiency per watt

3.2 소프트웨어 특성

3.2.1 소프트웨어 지원

LLM 가속기에서 중요한 요소 중 하나는 기존 LLM 소프트웨어와의 호환성이다. LLM을 효과적으로 학습 및 추론하기 위해서는 기존 머신 러닝 및 딥러닝 프레임워크(PyTorch, TensorFlow, JAX 등)와의 원활한 연동이 필수적이며, 또한 vLLM[6], DeepSpeed[7] 등과 같은 LLM 추론 전용 프레임워크와도 호환되어야 한다. 이러한 소프트웨어 스택의 호환성이 부족할 경우, 개발자가 특정 가속기에 맞는 코드 및 모델 변환을 별도로 수행해야 하므로 LLM 가속기의 실질적인 활용성과 채택률이 낮아질 수 있다. 표 4는 각 LLM 가속기가 지원하는 프레임워크와 소프트웨어 기술 스택을 나타낸다. 표에서 △는 공식적으로 지원하지는 않으나, 가속기 제조사에서 포팅(Porting)을 통해 지원하는 경우이다.

NVIDIA H100[22]은 LLM 학습 및 추론을 위해 널리 사용되는 가속기 중 하나이며, 대부분의 기존 머신 러닝 및 딥러닝 프레임워크에서 공식적으로 지원된다. NVIDIA는 특히 TensorRT-LLM과 같은 전용 최적화 프레임워크를 제공하며, CUDA 및 cuDNN과 같은 소프트웨어 스택을 통해 연산 가속을 지원한다. 이를 통해 PyTorch, TensorFlow, JAX 등 주요 프레임워크에서 H100을 기본적으로 지원할 수 있도록 최적화되어 있으며, NVIDIA의 NCCL을 통해 다중 GPU 또는 가속기 간 통신을 최적화할 수 있다.

표 4. 대규모 언어 모델 가속기 소프트웨어 지원
Table 4. Software supports of LLM accelerator

Accelerator	Dedicated software		Common framework				
	Framework	SDK/Library	TensorFlow	PyTorch	JAX	vLLM	DeepSpeed
NVIDIA H100	TensorRT-LLM	CUDA/cuDNN	O	O	O	O	O
AMD MI300X	AMD ROCm	AMD ROCm	△	△	X	O	△ (MI200)
Cerebras WSE-2	Cerebras Inference	CSL	△	△	X	X	△
Google TPU v4	JAX	XLA	O	△	O	O	X
Graphcore IPU	Poplar	Poplar SDK	△	△	△	△	X
Groq LPU	GroqWare/ GroqFlow	LPU Inference Engine	△	△	X	X	X
Intel Gaudi3	Intel Gaudi Software Suite	HCCL	△	△	X	△	△ (Gaudi2)
SambaNova SN40L	SambaNova Suite	SambaNova SDK	△	△	X	X	X
Tenstorrent Grayskull	TT-Buda	TT-Metalium	△	△	X	X	X

AMD MI300X[23]는 ROCm 소프트웨어 스택을 활용하여 PyTorch 및 TensorFlow에서 가속을 지원하며, RCCL을 통해 다중 GPU/가속기 간 통신을 최적화한다.

Cerebras WSE-2[18]는 CS-2 시스템을 기반으로 자체적인 소프트웨어 스택을 제공하며, 그래프 컴파일러(Graph compiler) 및 가중치 스트리밍(Weight Streaming) 기술을 활용하여 모델을 최적화한다.

Google TPU v4[17]는 XLA 및 JAX를 기반으로 LLM 연산을 가속하며, TensorFlow와 연동되는 아키텍처를 제공한다.

Graphcore IPU[24]는 Poplar SDK를 사용하여 PyTorch 및 TensorFlow에서 모델을 실행할 수 있도록 변환 및 최적화한다.

Groq LPU[25]는 소프트웨어 정의 텐서 스트리밍 프로세싱(TSP) 아키텍처를 활용하여 PyTorch 및 ONNX 모델을 변환하여 실행할 수 있도록 한다.

Intel Gaudi 3[26]는 oneDNN(oneAPI Deep Neural Network Library) 및 OpenVINO를 활용하여 기존 모델을 변환할 수 있도록 한다.

SambaNova SN40L[19]는 SambaNova Suite (SambaStudio)를 바탕으로 학습된 모델의 미세 조정(Fine-tuning), 최적화를 할 수 있는 기능을 지원한다.

Tenstorrent Grayskull[27]은 RISC-V 기반의 AI 가속기용 소프트웨어 프레임워크를 활용하여 ONNX 모델을 변환 및 실행할 수 있도록 한다.

3.2.2 소프트웨어 지원에 따른 가속기 선택

대부분의 LLM 가속기는 연산을 최적화하기 위한 인스트럭션과 하드웨어 구조를 가지므로 기존의 모델을 하드웨어 구조에 최적화할 수 있도록 자체 컴파일러, 모델 변환기 등을 제공하고 있다. 따라서 LLM 가속기의 소프트웨어 호환성 문제를 해결하기 위해 일부 제조사는 기존 프레임워크의 소스 코드에 자사 하드웨어를 지원하는 컴파일러 및 런타임을 추가하여 배포하기도 한다.

또한, LLM 가속기에서 다양한 LLM을 지원하기 위해 각 제조사는 하드웨어에 맞게 모델을 최적화하여 배포하지만, 모든 모델을 동일하게 지원하기는 어렵다. 이는 가속기의 연산 구조와 메모리 아키텍처에 따라 최적화가 필요한 모델이 다를 수 있으며, 정확성(Accuracy) 및 성능을 유지하면서도 최적화할 수 있는 범위가 제한적일 수 있기 때문이다.

따라서 LLM 가속기를 사용할 때는 가속기가 지원하는 프레임워크 및 모델 변환 방식, 소프트웨어 최적화 기법을 충분히 고려하여 선택하는 것이 중요하다.

기존의 합성곱 신경망(CNN, Convolutional Neural Network)의 학습과 추론에는 TensorFlow와 PyTorch 같은 범용 프레임워크들이 주로 사용되었다. 하지만 LLM의 경우 학습의 어려움으로 인해 사전에 학습된 파운데이션(Foundation) 모델을 사용해 추론하는 것이 일반적이며, 큰 모델 규모로 인해 범용 프레임워크보다는 추론 최적화 기능을 제공하는 LLM 엔진의 사용이 증가하고 있다. 대규모 LLM 서비스를 위해서는 vLLM[6], DeepSpeed[7]와 같이 PagedAttention, ZeRO Inference 등의 추론 최적화 기술과 다중 노드 환경에서의 분산 추론, 병렬화의 지원이 요구된다. 따라서 이러한 추론 엔진에서 지원하는 하드웨어를 사용하는 것이 서비스 개발 측면에서 용이성을 확보할 수 있다.

Ⅳ. 대규모 언어 모델 가속기 고찰 및 개발 방향

LLM 가속기의 핵심 역할은 대규모 병렬 연산, 고속 메모리 접근, 전력 효율 향상, 확장성 지원 등을 통해 학습 및 추론 성능을 최대화하는 것이다. 특히 LLM 연산은 대규모 행렬 곱셈과 비선형 변환을 반복 수행하므로 텐서 연산 최적화, 혼합 정밀도 처리, 효과적인 캐시, 메모리 계층 설계 등이 필수적이다. 또한, 초거대 모델 추론 시 대기 시간 최소화와 여러 가속기를 스케일아웃(Scale-out) 방식으로 병렬 연결할 수 있는 인터커넥션 기술도 매우 중요하다. 다음은 LLM 가속기 개발에서 주요하게 고려해야 할 사항들이다:

4.1 높은 연산 성능 및 병렬 처리 지원

LLM은 대규모 행렬 연산 및 텐서 연산을 반복적으로 수행하므로, 이를 효율적으로 처리하기 위해서는 대규모 병렬 처리가 가능한 가속기 설계가 필수적이다.

예를 들어 NVIDIA GPU는 수천 개의 스트리밍 멀티프로세서(SM, Streaming Multiprocessor)를 활용해 텐서 연산을 병렬 처리하며 최신 GPU에서는 텐서 코어나 행렬 곱셈 전용 유닛(MAC, Matrix Multiply Unit)을 이용해 LLM 연산을 최적화한다. 반면 FPGA 및 ASIC 기반 가속기는 연산 유닛과 데이터 흐름을 더욱 정밀하게 최적화함으로써 GPU보다 높은 연산 효율을 달성하기도 한다.

또한, LLM 파라미터 수가 계속 증가함에 따라 단일 하드웨어에서 학습 및 추론을 진행하기가 어려워지고 있다. Llama 3.1 및 3.2 모델[28]은 최대 4,050억 개의 파라미터를 가지며, 이를 32비트 정밀도 기준으로 처리하려면 약 1,944GB의 메모리가 필요하다(FP16은 972GB). 특히, 최근 공개된 Llama 4의 경우 전문가 혼합(MoE, Mixture of Experts) 모델로 최대 2조 개의 파라미터로 구성된다[29]. 이처럼 막대한 메모리 요구량을 충족하기 위해서는 다수 장치를 병렬로 연결해 연산 부하를 분산하는 파이프라인 병렬화(Pipeline parallelism)[30], 텐서 병렬화(Tensor parallelism)[31] 등의 기법이 필수적이다. 이러한 병렬화를 지원하기 위해서는 하드웨어 구조, 인터커넥션, 소프트웨어 최적화가 모두 이루어져야 한다.

4.2 복잡한 LLM 응용 최적화 지원

LLM이 단순한 텍스트 생성 이외에 최근에는 사고 과정을 단계별로 전개하거나 스스로 결과를 검증해 답변의 품질을 높이는 추론(Reasoning) 중심 테스트 타임 스케일링(CoT, Chain-of-Thought[32], Test Time Reasoning[33] 등) 기법과 계획을 세우고 연속적으로 작업을 수행하는 LLM 기반 AI 에이전트(Agent Laboratory[34] 등)가 주목받고 있다. 이러한 방법은 한 질문에 대해 모델을 여러 차례 호출하므로 지연 시간과 연산 비용이 급격히 증가한다. 실제 서비스에서 이를 감당하려면, 반복 호출에 적합한 모델, 시스템 최적화는 물론, 연산 속도를 높이는 하드웨어 가속 기술이 필수적이다. 특히 반복 호출 과정에서 필요한 데이터를 저장할 대용량 메모리도 함께 요구된다. LLM 가속기를 활용하면 대량의 추론 연산을 단축

해 사용자 체감 지연을 줄일 수 있으며, HBM와 같은 메모리를 탑재해 메모리 요구사항도 만족할 수 있다.

4.3 대용량 고대역폭 메모리 지원

LLM은 수십억~수천억 개 이상의 파라미터를 포함하며, 학습 및 추론 과정에서 매우 높은 메모리 대역폭과 용량이 요구된다.

최신 GPU 가속기는 HBM을 채택해 3TB/s 이상의 대역폭(NVIDIA H100의 경우 약 3.9TB/s)을 제공함으로써 모델의 메모리 병목을 완화하고 데이터 접근 속도를 극대화한다. FPGA, ASIC 기반 가속기 역시 SRAM, LPDDR, GDDR6 등 다양한 메모리 계층을 구성해 연산 효율을 높이고 있으며, 메모리 접근 시간을 줄이기 위한 PIM(Processing-in-Memory) [35] 등의 하드웨어 구조도 연구되고 있다.

LLM 추론 시, 쿼리(Query), 키(Key), 값(Value) 같은 내부 연산 정보를 효율적으로 적재, 관리할 수 있는 메모리 계층 설계가 중요하며, 메모리 프리페칭(Prefetching), 캐싱 등을 적용해 메모리 접근 시간을 줄여야 한다[6][36].

4.4 낮은 전력 소모 및 효율적인 연산

LLM 학습 혹은 추론 시 전력 효율을 높이는 하드웨어 설계는 비용 절감과 지속 가능성 측면에서 매우 중요하다.

GPU 가속기는 FP16, INT8 등 혼합 정밀도 연산을 지원해 연산 효율을 높이고 전력 소모를 줄이며, 전용 연산 유닛을 통해 데이터 이동을 최소화할 수 있다. ASIC 기반 가속기는 LLM 연산에 맞춤화된 블록을 설계하여 GPU, FPGA 대비 전력 소모를 더 크게 줄일 수 있고, 특히 데이터 이동을 최소화하는 구조를 채택함으로써 성능을 극대화한다. PIM 반도체를 활용해 메모리 내에서 직접 연산을 수행하는 방식도 데이터 이동에 따른 전력 소모를 줄이는 방법으로 주목받고 있다[35].

데이터센터와 클라우드 환경에서 LLM을 운영할 경우, 전력, 냉각 비용이 전체 운영 비용의 상당 부분을 차지하므로 저전력 및 고성능 가속기 설계가 더욱 중요하다. 최신 GPU나 ASIC 기반 LLM 가속기의 전력 소모 범위는 192W부터 최대 20kW까지 다양하며, 수천 개의 가속기를 연결하는 대규모 데이터센터에서는 전력 관리 및 최적 배치 전략이 필수적이다.

4.5 고속 인터커넥션 및 확장성 지원

최신 LLM 모델을 단일 가속기로 처리하기 어렵기 때문에, 여러 가속기를 병렬 연결해 확장(Scalability)하는 능력이 매우 중요하다.

GPU는 NVLink, NVSwitch, InfiniBand, Infinity Fabric 등 다양한 인터커넥션 기술을 통해 고속 통신을 지원하며, ASIC, FPGA 기반 가속기 역시 전용 인터페이스를 활용하는 초고속 데이터 전송 방식을 적용하고 있다. 예를 들어 Google TPU v4는 ICI로 대규모 클러스터 구성을 지원한다. RoCE 등의 네트워크 최적화 기술도 대규모 LLM 학습을 위해 연구되고 있다. 실제로 Llama 3 405B 모델 학습에는 16,000개 이상의 NVIDIA H100[22] 가속기가 사용되었는데[28], 이는 높은 확장성과 네트워크 인터커넥션 기술의 중요성을 단적으로 보여준다.

4.6 혼합 정밀도 및 모델 최적화 지원

LLM 학습 및 추론에서 연산 속도와 메모리 사용량을 동시에 개선하기 위해 혼합 정밀도와 양자화[37][38] 기법이 필수적이다.

GPU, ASIC 가속기는 FP16, INT8, INT4 등 다양한 정밀도를 지원하여 모델 연산 속도를 최적화하고 메모리 요구량을 줄인다. NVIDIA의 TensorRT-LLM, Google TPU의 XLA, AMD ROCm 등 소프트웨어 스택은 정밀도 변환을 자동 수행하는 기능을 제공한다. 이 과정에서 FP32-FP16 변환 시 오차를 최소화하고 양자화된 데이터를 복원하는 과정에서도 연산 성능을 유지해야 한다. LLM 프레임워크(vLLM[6], DeepSpeed[7] 등)는 이러한 하드웨어 친화적 모델 변환 기능을 지원해 성능과 메모리 균형을 목표로 한다.

4.7 소프트웨어 스택 및 개발자 친화적 환경

효율적인 LLM 가속을 위해서는 하드웨어뿐만 아니라 소프트웨어 스택 최적화가 필수적이다. 즉, PyTorch, TensorFlow, JAX 같은 프레임워크와 원활히 호환되도록 가속기를 설계하고, MLIR[39], Triton[40] 등 범용 컴파일러를 지원해 하드웨어별 최적화를 자동 수행하는 기능이 중요하다.

NVIDIA, AMD, Google 등은 각각 CUDA, ROCm, XLA 등 최적화 라이브러리를 제공하여 모델 학습과 추론 성능을 높이고 있으며, 최근 널리 쓰이는 LLM 추론 프레임워크와의 호환성 확보도 가속기 개발 시 고려해야 한다.

V. 결 론

본 논문에서는 LLM의 학습 및 추론을 가속하기 위한 전용 하드웨어의 필요성을 살펴보고, 주요 상용 LLM 가속기의 하드웨어 및 소프트웨어 특성을 종합적으로 분석하였다. 특히 NVIDIA H100, AMD MI300X, Cerebras WSE-2, Google TPU v4, Graphcore IPU, Groq LPU, Intel Gaudi 3, SambaNova SN40L, Tenstorrent Grayskull과 같은 다양한 ASIC 기반 가속기들의 아키텍처와 메모리, 성능, 인터커넥션 기술, 소프트웨어 호환성 등을 비교함으로써 LLM 가속기의 설계 방향과 기술적 과제를 논의하였다.

LLM 가속기의 효율적인 활용을 위해 다음과 같은 사항이 중요함을 확인할 수 있었다. 첫째, 모델 규모가 커짐에 따라 단일 가속기의 한계를 극복하기 위한 다중 가속기 연결 기술과 고속 인터커넥션이 필수적이다. 둘째, 연산 성능과 에너지 효율을 동시에 극대화하기 위해 혼합 정밀도 및 양자화 기법을 하드웨어에서 지원하고, 이를 자동으로 최적화하는 소프트웨어 스택이 필요하다. 셋째, HBM, SRAM, LPDDR 등 다양한 메모리 방식의 장단점을 고려해 메모리 대역폭과 용량을 균형 있게 확보하는 전략이 요구된다. 마지막으로, LLM 가속기가 제공하는 고유한 연산 구조를 활용하기 위해서는 기존 PyTorch, TensorFlow, JAX, vLLM, DeepSpeed 등의 프레임워크와 원활하게 연동할 수 있는 개발 환경을 구축해야 한다.

향후 연구에서는 실제 서비스 워크로드와 벤치마크 테스트 결과를 활용해 가속기별 성능, 전력 효율 및 발열 등의 특성을 비교, 분석하며, 모델 구조별 최적의 하드웨어와 소프트웨어 구성에 대한 분석을 수행한다. 이를 위해서 각 가속기의 세부 구조와 하드웨어 최적화 방법에 대한 분석과 실제 사용 가능한 환경을 구축한다. 가속기별 성능 분석을 위해 모듈별(어텐션, Matmul 등) 성능과 가속기에서 지원하는 LLM을 바탕으로 전체 모델의 지연 시간, 토큰 생성 관련 성능(TTFT, Time to First Token, TBT, Time Between Tokens 등)을 측정하고 비교 분석할 계획이다.

References

[1] OpenAI, "GPT-4 Technical Report", arXiv preprint arXiv:2303.08774, pp. 1-100, Mar. 2023. https://doi.org/10.48550/arXiv.2303.08774.

[2] M. Xu, et al., "A Survey of Resource-Efficient LLM and Multimodal Foundation Models", arXiv preprint arXiv:2401.08092, pp. 1-65, Jan. 2024. https://doi.org/10.48550/arXiv.2401.08092.

[3] Y. Huang, et al., "New Solutions on LLM Acceleration, Optimization, and Application", Proc. of the 61st ACM/IEEE Design Automation Conference, San Francisco, CA, USA, pp. 1-4, Jun. 2024. https://doi.org/10.1145/3649329.3663517.

[4] DeepSeek-AI, "DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning", arXiv preprint arXiv:2501.12948, pp. 1-22, Jan. 2025. https://doi.org/10.48550/arXiv.2501.12948.

[5] A. Yehudai, et al., "Survey on Evaluation of LLM-based Agents", arXiv preprint arXiv:2503.16416, pp. 1-20, Mar. 2025. https://doi.org/10.48550/arXiv.2503.16416.

[6] W. Kwon, et al., "Efficient Memory Management for Large Language Model Serving with PagedAttention", Proc. of the 29th Symposium on Operating Systems Principles, Koblenz, Germany, pp. 611-626, Oct. 2023. https://doi.org/10.1145/3600006.3613165.

[7] R. Y. Aminabadi, et al., "DeepSpeed-Inference: Enabling Efficient Inference of Transformer Models at Unprecedented Scale", SC22: Internation al Conference for High Performance Computing, Networking, Storage and Analysis, Dallas, TX, USA, pp. 1-15, Nov. 2022. https://doi.org/10.1109/SC41404.2022.00051.

[8] X. Zhu, J. Li, Y. Liu, C. Ma, and W. Wang, "A Survey on Model Compression for Large Language Models", Transactions of the Association for Computational Linguistics, Vol. 12, pp. 1556-1577, Nov. 2024. https://doi.org/10.1162/tacl_a_00704.

[9] Z. Wan, et al., "Efficient Large Language Models: A Survey", arXiv preprint arXiv:2312.03863, pp. 1-67, Dec. 2023. https://doi.org/10.48550/arXiv.2312.03863.

[10] K. T. Chitty-Venkata, S. Mittal, M. Emani, V. Vishwanath, and A. K. Somani, "A Survey of Techniques for Optimizing Transformer Inference", Journal of Systems Architecture, Vol. 144, pp. 102990, Nov. 2023. https://doi.org/10.1016/j.sysarc.2023.102990.

[11] S. Zeng, et al., "FlightLLM: Efficient Large Language Model Inference with a Complete Mapping Flow on FPGAs", FPGA '24: Proc. of the 2024 ACM/SIGDA International Symposium on Field Programmable Gate Arrays, Monterey, CA, USA, pp. 223-234, Mar. 2024. https://doi.org/10.1145/3626202.3637562.

[12] M. Huang, A. Shen, K. Li, H. Peng, B. Li, Y. Su, and H. Yu, "EdgeLLM: A Highly Efficient CPU-FPGA Heterogeneous Edge Accelerator for Large Language Models", IEEE Transactions on Circuits and Systems I: Regular Papers, No. 99, pp. 1-14, Mar. 2025. https://doi.org/10.1109/TCSI.2025.3546256.

[13] E. Frantar, R. L. Castro, J. Chen, T. Hoefler, and D. Alistarh, "Marlin: Mixed-Precision Auto-Regressive Parallel Inference on Large Language Models", Proc. of the 30th ACM SIGPLAN Annual Symposium on Principles and Practice of Parallel Programming, Las Vegas, NV, USA, pp. 239-251, Mar. 2025. https://doi.org/10.1145/3710848.3710871.

[14] Z. Guan, H. Huang, Y. Su, H. Huang, N. Wong, and H. Yu, "APTQ: Attention-Aware Post-Training Mixed-Precision Quantization for Large Language Models", Proc. of the 61st ACM/IEEE Design Automation Conference, San Francisco, CA, USA, pp. 1-6, Jun. 2024. https://doi.org/10.1145/3649329.3658498.

[15] H. Chen, J. Zhang, Y. Du, S. Xiang, Z. Yue, N. Zhang, Y. Cai, and Z. Zhang, "Understanding the Potential of FPGA-Based Spatial Acceleration for Large Language Model Inference", ACM Transactions on Reconfigurable Technology and Systems, Vol. 18, No. 1, pp. 1-29, 2024. https://doi.org/10.1145/3656177.

[16] A. He, D. Key, M. Bulling, A. Chang, S. Shapiro, and E. Lee, "HLSTransform: Energy-Efficient Llama 2 Inference on FPGAs Via High Level Synthesis", arXiv preprint arXiv:2405.00738, pp. 1-9, Apr. 2024. https://doi.org/10.48550/arXiv.2405.00738.

[17] N. Jouppi, et al., "TPU v4: An Optically Reconfigurable Supercomputer for Machine Learning with Hardware Support for Embeddings", Proc. of the 50th Annual International Symposium on Computer Architecture, Orlando, FL, USA, pp. 1-14, Jun. 2023. https://doi.org/10.1145/3579371.3589350.

[18] S. Lie, "Cerebras Architecture Deep Dive: First Look Inside the Hardware/Software Co-Design for Deep Learning", IEEE Micro, Vol. 43, No. 3, pp. 18-30, May 2023. https://doi.org/10.1109/MM.2023.3256384.

[19] R. Prabhakar, et al., "16.4: SambaNova SN40L: A 5nm 2.5D Dataflow Accelerator with Three Memory Tiers for Trillion Parameter AI", 2025 IEEE International Solid-State Circuits Conference (ISSCC), San Francisco, CA, USA, Vol. 68, pp. 288-290, Feb. 2025. https://doi.org/10.1109/ISSCC

49661.2025.10904578.

[20] Y. Fu, "Challenges in Deploying Long-Context Transformers: A Theoretical Peak Performance Analysis", arXiv preprint arXiv:2405.08944, pp. 1-11, May 2024. https://doi.org/10.48550/arXiv.2405.08944.

[21] K. Kim and M. Park, "Present and Future, Challenges of High Bandwidth Memory (HBM)", 2024 IEEE International Memory Workshop (IMW), Seoul, Korea, pp. 1-4, May 2024. https://doi.org/10.1109/IMW59701.2024.10536972.

[22] J. Choquette, "NVIDIA Hopper H100 GPU: Scaling Performance", IEEE Micro, Vol. 43, No. 3, pp. 9-17, May 2023. https://doi.org/10.1109/MM.2023.3256796.

[23] A. Smith, et al., "11.1 AMD InstinctTM MI300 Series Modular Chiplet Package—HPC and AI Accelerator for Exa-Class Systems", 2024 IEEE International Solid-State Circuits Conference (ISSCC), San Francisco, CA, USA, Vol. 67, pp. 490-492, Feb. 2024. https://doi.org/10.1109/ISSCC49657.2024.10454441.

[24] S. Knowles, "Graphcore", 2021 IEEE Hot Chips 33 Symposium (HCS), Palo Alto, CA, USA, pp. 1-25, Aug. 2021. https://doi.org/10.1109/HCS52781.2021.9567075.

[25] D. Abts, et al., "The Groq Software-Defined Scale-Out Tensor Streaming Multiprocessor: From Chips-to-Systems Architectural Overview", 2022 IEEE Hot Chips 34 Symposium (HCS), Cupertino, CA, USA, pp. 1-69, Aug. 2022. https://doi.org/10.1109/HCS55958.2022.9895630.

[26] R. Kaplan, "Intel Gaudi 3 AI Accelerator: Architected for Gen AI Training and Inference", 2024 IEEE Hot Chips 36 Symposium (HCS), pp. 1-16, Aug. 2024. https://doi.org/10.1109/HCS61935.2024.10665178.

[27] N. Brown and R. Barton, "Accelerating Stencils on the Tenstorrent Grayskull RISC-V Accelerator", SC24-W: Workshops of the International Conference for High Performance Computing, Networking, Storage and Analysis, Atlanta, GA, USA, pp. 1690-1700, Nov. 2024. https://doi.org/10.1109/SCW63240.2024.00211.

[28] Meta, "The Llama 3 Herd of Models", arXiv preprint arXiv:2407.21783, pp. 1-92, Jul. 2024. https://doi.org/10.48550/arXiv.2407.21783.

[29] Meta, "The Llama 4 herd: The beginning of a new era of natively multimodal AI innovation", https://ai.meta.com/blog/llama-4-multimodal-intelligence/. [accessed: Apr. 21, 2025].

[30] Z. Jiang, et al., "MegaScale: Scaling Large Language Model Training to More Than 10,000 GPUs", 21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24), Santa Clara, CA, USA, pp. 745-760, Apr. 2024.

[31] R. B. Prabhakar, H. Zhang, and D. Wentzlaff, "Kraken: Inherently Parallel Transformers for Efficient Multi-Device Inference", Advances in Neural Information Processing Systems, Vol. 37, pp. 7957-7980, Dec. 2024.

[32] N. Muennighoff, et al., "s1: Simple test-time scaling", arXiv preprint arXiv:2501.19393, pp. 1-46, Jan. 2025. https://doi.org/10.48550/arXiv.2501.19393.

[33] X. Zhang, C. Du, T. Pang, Q. Liu, W. Gao, and M. Lin, "Chain of Preference Optimization: Improving Chain-of-Thought Reasoning in LLMs", Advances in Neural Information Processing Systems, Vol. 37, pp. 333-356, Dec. 2024.

[34] S. Schmidgall, Y. Su, Z. Wang, X. Sun, J. Wu, X. Yu, J. Liu, Z. Liu, and E. Barsoum, "Agent Laboratory: Using LLM Agents as Research Assistants", arXiv preprint arXiv:2501.04227, pp. 1-56, Jan. 2025. https://doi.org/10.48550/arXiv.2501.04227.

[35] Y. Gu, A. Khadem, S. Umesh, N. Liang, X. Servot, O. Mutlu, R. Iyer, and R. Das, "PIM Is All You Need: A CXL-Enabled GPU-Free System for Large Language Model Inference", arXiv preprint arXiv:2502.07578, pp. 1-20, Feb. 2025. https://doi.org/10.48550/arXiv.2502.07578.

[36] B. Gao, et al., "Cost-Efficient Large Language Model Serving for Multi-Turn Conversations with CachedAttention", 2024 USENIX Annual Technical Conference (USENIX ATC 24), Santa Clara, CA, USA, pp. 111-126, Jul. 2024.

[37] Q. Li, et al., "FPTQ: Fine-Grained Post-Training Quantization for Large Language Models", arXiv preprint arXiv:2308.15987, pp. 1-17, Aug. 2023. https://doi.org/10.48550/arXiv.2308.15987.

[38] C. Lee, J. Jin, T. Kim, H. Kim, and E. Park, "OWQ: Outlier-Aware Weight Quantization for Efficient Fine-Tuning and Inference of Large Language Models", Proc. of the AAAI Conference on Artificial Intelligence, Vancouver, Canada, Vol. 38, No. 12, pp. 13355-13364, Feb. 2024. https://doi.org/10.1609/aaai.v38i12.29237.

[39] C. Lattner, et al., "MLIR: A Compiler Infrastructure for the End of Moore's Law", arXiv preprint arXiv:2002.11054, pp. 1-21, Feb. 2020. https://doi.org/10.48550/arXiv.2002.11054.

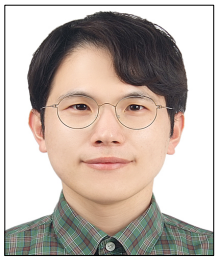
[40] P. Tillet, H.-T. Kung, and D. Cox, "Triton: An Intermediate Language and Compiler for Tiled Neural Network Computations", Proc. of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages, Phoenix, AZ, USA, pp. 10-19, Jun. 2019. https://doi.org/10.1145/3315508.3329973.

저자소개

박 시 형 (Sihyeong Park)

2014년 2월 : 충남대학교 컴퓨터공학과(공학사)
2016년 2월 : 충남대학교 컴퓨터공학과(공학석사)
2021년 2월 : 충남대학교 컴퓨터공학과(공학박사)
2021년 3월 ~ 현재 : 한국전자기술연구원 선임
관심분야 : 임베디드 시스템, 실시간 시스템, 인공지능

이 제 민 (Jemin Lee)



2011년 8월 : 충남대학교 컴퓨터공학과(공학사)
2017년 8월 : 충남대학교 컴퓨터공학과(공학박사)
2017년 9월 ~ 2018년 8월 : 한국과학기술원 박사후연구원
2018년 12월 ~ 현재 : 한국전자통신연구원 선임
2023년 9월 ~ 현재 : 한국과학기술연합대학원대학교 조교수
관심분야 : 모델 경량화, 컴파일러, 모바일 컴퓨팅

김 병 수 (Byung-Soo Kim)

2006년 2월 : 인하대학교 정보통신공학과(공학사)
2008년 2월 : 인하대학교 정보통신공학과(공학석사)
2013년 8월 : 인하대학교 정보통신공학과(공학박사)
2013년 8월 ~ 현재 : 한국전자기술연구원 센터장
관심분야 : 뉴로모픽 하드웨어, Processing-in-Memory(PIM), 양자 회로 설계

전 석 훈 (Seokhun Jeon)

2012년 2월 : 숭실대학교 정보통신공학과(공학석사)
2013년 1월 ~ 2017년 3월 : 씨게이트 코리아 디자인센터 선임
2017년 12월 ~ 현재 : 한국전자기술연구원 선임
관심분야 : 인공지능 성능 최적화, PIM 기반 소프트웨어, 양자 컴퓨팅 시뮬레이션