서버리스 환경에서 동적 배칭 기반 AI 추론 지연 성능 분석

김태림[1], 김예진[2], 김건민[3], 김경백[3]
[1]전남대학교 인공지능학부 학부생
[2]전남대학교 소프트웨어공학과 학부생
[3]전남대학교 인공지능융합학과
ktr0706@jnu.ac.kr, ye031010@jnu.ac.kr, geonminkim@jnu.ac.kr, kyungbaekkim@jnu.ac.kr,

Performance Analysis of AI Inference Latency with Dynamic Batching in a Serverless Environment

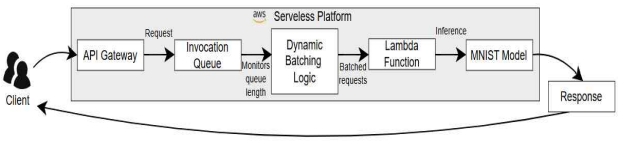
Taerim Kim[1], Yejin Kim[2], Geonmin Kim[3], Kyungbaek Kim[3]
[1]Dept. of Artificial Inelligence, Chonnam National University
[2]Dept. of Software Engineering, Chonnam National University
[3]Dept. of AI Convergence, Chonnam National University

요 약

서버리스 환경은 자동 확장과 종량제 과금, 이벤트 기반 실행을 제공하나 AI 추론 시 지연이 발생할 수 있다. 본 연구는 정적 배칭과 동적 배칭을 적용하여 AWS Lambda 기반 MNIST 모델에서 균일·버스티 요청 환경의 평균/p50/p99 지연과 SLO 충족률을 분석하였다. 실험 결과 정적 배칭 대비 동적 배칭은 지연을 감소시키고 SLO 충족률을 향상시키며 비용 효율성을 높이는 효과를 보였다.

1. 서론

서버리스 컴퓨팅은 개발자가 서버 인프라를 직접 관리하지 않고 애플리케이션을 빌드 및 실행할 수 있도록 지원하는 클라우드 네이티브 모델로 서버리스의 특성은 AI 추론과 같이 요청 패턴이 불규칙하고 짧은 응답 시간이 중요한 워크로드에 적합하다[1]. 그러나 서버리스 환경에서 대규모 AI 모델이나 버스티 워크로드의 지연이 증가할 수 있다. 본 연구는 동적 배칭을 통해 서버리스 기반 AI 추론 시스템의 지연 특성을 분석하고, 균일·버스티 요청과 같은 다양한 워크로드 패턴에서 서비스 수준 목표 충족 여부를 평가한다. 정적 배칭과 대비해 동적 배칭의 성능 특성을 분석하고, 서버리스 환경에서 효율적인 AI 추론 설계 방향을 제시한다.



(그림 1) 서버리스 환경에서 동적 배칭 기반 추론 구조

2. 이론적 배경 및 관련 연구

2.1 동적 배칭 기법

본 연구에서는 서버리스 환경에서의 요청 처리 효율을 분석하기 위해 동적 배칭(Dynamic Batching) 기법을 적용하였다. 동적 배칭은 개별 요청을 바로 처리하지 않고 큐(queue)에 일정 시간 동안 모아두었다가 조건이 충족되면 묶어서(batch) 처리하는 방식이다. 이를 위해 두 가지 기준을 설정하였다. 첫째, 큐에 요청이 5개 이상 쌓이면 즉시 배치를 전송한다. 둘째, 임계값에 도달하지 못하더라도 0.3초가 지나면 현재까지 모인 요청을 묶어 전송한다. 이러한 방식은 짧은 시간 내 다수의 요청이 집중될 경우 효율적으로 처리량을 높이고, 요청이 드물게 발생하는 상황에서는 대기 시간을 최소화하여 지연을 줄일 수 있어 처리 효율성과 응답 성능 간 균형을 유지한다.

2.2 기존 연구 동향

기존 연구는 MBS와 같은 기법을 통해 처리량과 비용을 최적화하였으나 패딩 오버헤드 및 이질적 요청 처리에서 한계가 보고되었다[2]. 또한, 모델 분할 기반 접근은 통신 지연을 증가시키고, Knative·OpenFaaS 평가 연구는 플랫폼 특성에 따른 성능 차이를 보였다[3]. 최근 연구는 특정 모델이나 환경에 한정되어 있으며 다양한 요청 패턴과 실제 워크로드에 대한 정량적 성능 분석은 제한적이다[4].

배치 방식	워크로드	평균/p50/p99 지연(ms)	SLO 충족률(%)	Init Duration(ms)	비용($)
정적배칭	균일	168.9/159.4/248.9	94.5	8250.38	0.000679
	버스티	164.5/133.4/432.2	94.8	6416.08	0.000739
동적배칭	균일	**133.0/126.6/217.7**	**98.75**	**4369.02**	**0.000609**
	버스티	**139.0/127.0/208.0**	**98.95**	**4842.20**	**0.000620**

(표 1) 서버리스 환경에서 정적 및 동적 배칭 방식의 MNIST 추론 성능 비교

3. 실험 설정 및 방법

본 연구는 서버리스 환경에서 동적 배칭이 AI 추론 지연에 미치는 영향을 평가하기 위해 AWS Lambda와 Docker 이미지로 배포된 MNIST 분류 모델을 사용하였다. Lambda 함수는 1024MB 메모리와 60초 타임아웃으로 설정하였다. 동적 배칭은 함수 호출 큐의 길이를 모니터링하여 배치 크기를 조절하며 요청이 적을 때는 지연 최소화, 요청이 급증할 때는 처리량 극대화를 목표로 한다. 워크로드는 균일 요청과 버스티 요청 두 가지 유형을 사용하였고, 평균/p50/p99 지연, SLO 충족률(200ms 기준), Init Duration과 비용을 측정하였다.

4. 실험 결과

(표 1)에 제시된 것과 같이 동적 배칭을 적용한 경우 정적 배칭 대비 전반적으로 추론 성능 개선이 됨을 알 수 있다. 특히 버스티 워크로드에서 평균 지연은 164.5ms에서 139.0ms로 약 15% 감소하였으며 p99 지연 역시 432.2ms에서 208.0ms로 절반 이상 줄어들어 지연 편차가 완화되는 효과를 확인하였다. SLO 충족률 또한 버스티 요청에서 94.8%에서 98.95%로 향상되었다. 균일 워크로드에서도 평균 지연은 168.9ms에서 133.0ms로 약 21% 줄었고, p50 지연 역시 159.4ms에서 126.6ms로 감소하였다. 결과적으로 두 워크로드 모두에서 SLO 충족률이 94%대에서 98% 이상으로 향상되었다. 또한, 동적 배칭 적용 시 Init Duration이 감소하는 경향을 확인할 수 있었다. 이는 배치 처리로 초기화 비용이 분산되면서 일부 요청에서 초기화 지연이 완화된 결과로 해석된다. 비용 측면에서도 동적 배칭 적용 시 소폭 절감되는 경향을 확인할 수 있었다. 이는 지연 최적화뿐만 아니라 비용 효율성 측면에서도 동적 배칭이 효과적임을 보여준다.

5. 결론 및 향후 연구

본 연구에서는 서버리스 환경에서 정적 배칭과 동적 배칭의 성능을 비교하여 배칭 방식이 AI 추론 지연에 미치는 영향을 정량적으로 분석하였다. 정적 배칭은 고정된 배치 크기로 단순하지만, 요청량 변동 시 지연이 증가할 수 있다. 반면 동적 배칭은 큐 길이에 따라 배치 크기를 조정해 자원 활용과 지연을 개선한다. 실험 결과 동적 배칭 적용 시 평균/p50/p99 지연이 감소하고 SLO 충족률이 향상되어 안정적 추론 성능을 확인하였다.

향후 연구에서는 동적 배칭과 콜드 스타트 완화를 결합하여 추론 지연 및 초기화 지연을 동시에 최소화하는 방안을 연구하고, 더 복잡한 AI 모델과 다양한 서버리스 플랫폼에서 동적 배칭 성능을 비교 평가함으로써 일반화 가능성을 검증할 수 있다.

Acknowledgement

본 연구는 2025년도 과학기술정보통신부 및 정보통신기획평가원의 소프트웨어중심대학사업의 연구결과로 수행되었습니다.(2021-0-01409)(34%)이 논문은 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원-지역지능화혁신인재양성사업의 지원을 받아 수행된 연구임(IITP-2025-RS-2022-00156287)(33%)본 연구성과는 2025년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(RS-2025-25398164)(33%)

참고문헌

[1] L. Wang, Y. Jiang, and N. Mi, "Advancing Serverless Computing for Scalable AI Model Inference: Challenges and Opportunities," WoSC10 '24, 2024, pp. 1-2.

[2] A. Ali, R. Pinciroli, F. Yan, and E. Smirni, "Optimizing Inference Serving on Serverless Platforms," Proceedings of the VLDB Endowment, vol. 15, no. 10, 2022.

[3] A. Barrak, F. Petrillo, and F. Jaafar, "Serverless on Machine Learning: A Systematic Mapping Study," IEEE Access, vol. 10, 2022.

[4] A. Bhattacharjee, A. D. Chhokra, Z. Kang, H. Sun, A. Gokhale, and G. Karsai, "BARISTA: Efficient and Scalable Serverless Serving System for Deep Learning Prediction Services," 2019 IEEE International Conference on Cloud Engineering (IC2E), Prague, Czech Republic, 2019