모빌리티 환경을 위한 AI 모델 분할 기반 지능형 컨테이너 오케스트레이션

박기철, 안재훈, 김영환
한국전자기술연구원
{kcpark, corehun, yhkim93}@keti.re.kr

Intelligent Container Orchestration Based on AI Model Partitioning for Mobility Environments

Kicheol Park, Jaehoon An, Younghwan Kim
Korea Electronics Technology Institute

요 약

본 논문은 모빌리티 서비스는 자율주행, 스마트 교통, 도심 항공 모빌리티(UAM) 등 다양한 형태로 발전하고 있으며, 실시간 인지·판단을 위한 AI 모델의 성능과 효율성이 핵심 경쟁 요소가 되고 있다. 그러나 대규모 AI 모델을 차량이나 모빌리티 디바이스에서 전부 실행하는 것은 연산 부담과 지연 문제를 야기한다. 본 논문에서는 AI 모델을 Backbone과 Neck+Head로 분할하여, 모빌리티 디바이스와 엣지 서버에 적절히 배치함으로써 처리 지연을 최소화하는 지능형 컨테이너 오케스트레이션 기법을 제안한다. 제안 기법은 모빌리티 환경에서 분할된 워크로드와 모빌리티 디바이스의 특성을 반영하여 최적의 실행 경로를 결정함으로써, 미션 수행에 지장을 주지 않으면서 안정적이고 효율적인 처리를 가능하게 한다.

Ⅰ. 서 론

모빌리티 환경에서는 자율주행 차량, 도심 항공 모빌리티(UAM)와 같이 다양한 형태의 이동체가 센서 데이터를 실시간 분석하여 주변 상황을 인지하고, 경로를 계획하며, 주행을 제어해야 한다. 이러한 과정에서 AI 모델은 높은 연산량과 짧은 응답시간이 동시에 요구되며, 특히 안전과 직결되는 작업에서는 성능 저하 없이 즉시 처리하는 것이 필수적이다[1].

그러나 고성능 AI 모델을 모빌리티 디바이스에서 전부 실행하는 경우, 연산 자원 부족과 전력 제약, 네트워크 통신 지연 등의 문제로 안정적인 성능을 유지하기 어렵다. 이를 해결하기 위해 SDI(Software-Defined Infrastructure)와 엣지 컴퓨팅을 결합하여, 컴퓨팅·네트워킹·스토리지 자원을 유연하게 관리하고, 단말과 인근 엣지 서버 간에 연산을 적절히 분배하여 처리 속도와 효율성을 향상시킬 수 있다[2].

특히 AI 모델을 계층별로 분할하여, Backbone 는 모빌리티 디바이스에서 수행하고, 후속 통합·추론 단계는 엣지 서버에서 실행하는 구조는 자원 부담을 분산하고 지연을 최소화하는 효과적인 방법이다[4].

본 논문에서는 SDI 기반 환경에서 AI 모델 분할 실행을 지원하는 지능형 컨테이너 오케스트레이션 기법을 제안한다. 제안 기법은 분할된 워크로드와 모빌리티 디바이스의 특성을 반영하여 최적의 실행 경로를 결정함으로써, 미션 수행에 지장을 주지 않으면서 안정적이고 효율적인 처리를 가능하게 한다.

Ⅱ. 본론

1. 모빌리티 환경에서의 AI 추론 특성 및 제약 요인

모빌리티 환경에서 제공되는 자율주행, 지능형 교통관리, 실시간 객체 인식 등의 AI 서비스는 초저지연성과 고신뢰성을 동시에 요구한다[2]. 차량·드론·로봇 등 이동형 디바이스는 카메라, 라이다(LiDAR), 레이더 등 복수 센서로부터 초당 수십~수백 프레임의 데이터를 수집하며, 이를 실시간 분석하기 위해 GPU/NPU와 같은 연산 자원이 필요하다[3].

본 논문에서는 대표적인 소형 모빌리티 플랫폼인 터틀봇3에 라즈베리파이5를 온보드 컴퓨팅 장치로 탑재한 환경을 모빌리티 디바이스로 정의하고 실험을 진행하였다. 이와 같은 소형 플랫폼은 연산·전력·발열·폼팩터 제약을 동시에 받으며, 특히 모델 학습까지 수행하기에는 연산 자원이 절대적으로 부족하다. 추론 역시 단말 CPU만으로는 높은 프레임레이트를 유지하기 어렵고, 이동에 따른 네트워크 품질 변동성으로 클라우드 단독 처리 시 일관된 지연 보장이 불가능하다[1].

이러한 제약을 완화하기 위해, AI 모델을 분할하여 온보드 디바이스(라즈베리파이5)와 외장 NPU(Hailo-8)에 연산을 분산(offload)하는 방식을 적용하였다. 컨볼루션 등 연산집약적인 처리를 NPU로 이전함으로써 추론 지연을 줄이고, 단말 CPU의 전력·열 부담을 경감하는 것을 목표로 한다.

2. SDI 기반 지능형 오케스트레이션 아키텍처

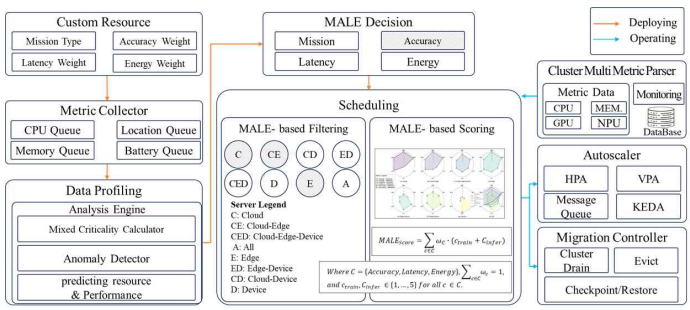

그림 1 모빌리티 환경을 위한 AI 모델 분할 기반 지능형 컨테이너 오케스트레이션 아키텍처

그림 1은 Mission, Accuracy, Latency, Energy(MALE) 지표를 기반으로 한 지능형 오케스트레이션 아키텍처를 보여준다. MALE은 미션 목표 달성을 위한 정확도, 지연 시간, 에너지 효율을 종합적으로 고려하여 최적의 자원 할당 및 컨테이너 배포를 수행한다. 이를 위해 시스템은 Kubernetes를 기반으로 컨테이너 오케스트레이션 환경을 구성하며, 다음과 같은 기능을 구현한다.

Data Profiling: 모델의 복잡도, 실행 환경의 이기종 자원 특성, 워크로드의 혼합 중요도(Mixed-Criticality)를 분석한다.

Metric Collector: CPU, 메모리, 배터리, 위치 등의 상태 데이터를 실시간으로 수집한다.

MALE Decision: 수집된 지표를 바탕으로 정확도, 지연 시간, 에너지 효율 간의 균형을 고려하여 스케줄링 기준을 결정한다.

Scheduler: 필터링과 점수화를 통해 컨테이너를 실행할 최적의 디바이스(Cloud, Edge, Mobility Device 등)를 선택한다.

Autoscaler & Migration Controller): 워크로드 변화에 따라 컨테이너 수를 자동 조정하고, 필요 시 체크포인트를 기반으로 컨테이너를 다른 노드로 이동시킨다.

3 모델 분할 기반 엣지-모빌리티 협력 구조

본 연구에서는 YOLOv5 계열의 객체 인식 모델을 대상으로, Backbone 단계와 Neck+Head 단계를 분리하는 계층 분할 방식을 채택하였다. Backbone 단계는 입력 영상으로부터 저차원에서 고차원에 이르는 특징(feature)을 추출하는 CNN 계층으로 구성되며, 연산량은 크지만 데이터 크기 축소 효과를 제공한다. 반면, Neck+Head 단계는 다중 스케일 feature fusion과 객체 검출을 수행하며, 높은 연산량과 메모리 접근이 요구된다.

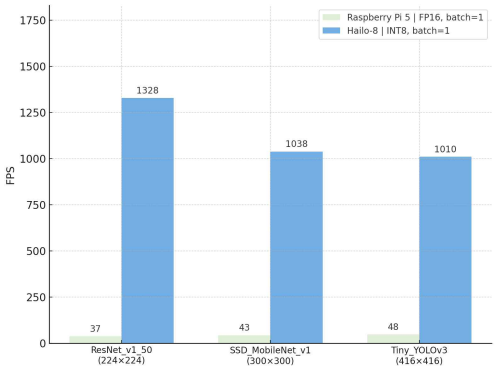

그림 2 라즈베리파이 5와 Hailo-8 가속기의 AI 모델별 FPS 성능 비교

그림 2는 라즈베리파이 5와 Hailo-8 가속기에서 다양한 AI 모델의 FPS 성능을 비교한 결과를 나타낸다. 실험 결과, 라즈베리파이 5 단독 환경에서는 ResNet, SSD-MobileNet, Tiny YOLOv3 등 상대적으로 경량화된 모델만 실시간 처리 성능을 확보할 수 있었으며, 이는 하드웨어 스펙상 복잡도가 높은 모델 운용에 제약이 있음을 시사한다.

이러한 한계를 극복하기 위해, 본 연구에서는 YOLOv5를 Backbone과 Neck+Head로 분할하고, 각 모듈을 컨테이너 단위로 패키징 하여 Kubernetes 환경에서 배포하는 방식을 제안한다. 그림 3은 이러한 방식을 통해 실행 위치가 변경되더라도 동일한 이미지를 재사용할 수 있으며, 환경 의존성을 최소화하면서 고성능 모델의 분산 처리가 가능함을 보여준다.

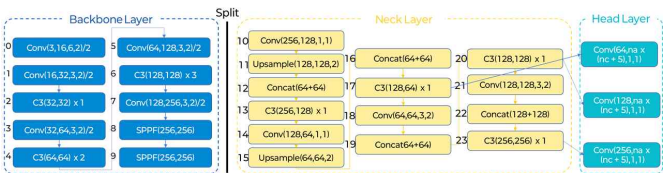

그림 3 : YOLOv5 모델 분할 테스트베드 Backbone - Neck - Head 계층 분할 지점

Ⅲ. 결론

본 논문에서는 AI 모델을 계층 단위로 분할하여 모빌리티 디바이스와 엣지 서버에 분산 배치하는 지능형 컨테이너 오케스트레이션 기법을 제안하였다. 초기 단계 연산은 모빌리티 디바이스에서, 후속 단계 연산은 엣지 서버에서 수행함으로써 연산 부하를 분산하고 지연을 최소화하였다. 제안 구조는 네트워크 상태와 시스템 자원 변동에 따라 실행 경로를 동적으로 조정할 수 있도록 설계되었으며, 실험 결과 모빌리티 디바이스 단독 처리 대비 지연 시간 단축과 처리 효율 향상이 기대된다. 향후에는 다양한 AI 모델과 서비스 환경으로의 확장 적용 및 실제 모빌리티 디바이스-엣지 테스트베드 기반 성능 검증을 수행할 예정이다.

ACKNOWLEDGMENT

이 논문은 2024년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구 결과임 (No.RS-2024-00406245, 미래 모빌리티를 위한 소프트웨어 정의형 인프라스트럭처 기술 개발)

참 고 문 헌

[1] W. Rafique, L. Qi, I. Yaqoob, et al., “Complementing IoT Services Through Software Defined Networking and Edge Computing: A Comprehensive Survey,” IEEE Communications Surveys & Tutorials, vol. 22, no. 3, pp. 1761 - 1804, 2020.

[2] K. C. Park, J. H. An, Y. H. Kim, H. Kim, “Scalable Orchestration Design for Mixed-Critical Task Scheduling in SDI-based Mobility Systems,” in Proc. ACM Symposium on Applied Computing (SAC), pp. 1407 - 1411, 2025.

[3] F. Lumpp, F. Fummi, H. D. Patel, N. Bombieri, “Enabling Kubernetes Orchestration of Mixed-Criticality Software for Autonomous Mobile Robots,” IEEE Transactions on Robotics, vol. 40, no. 2, pp. 540 - 553, 2024.

[4] J. C. Lee, Y. Kim, S. T. Moon, J. H. Ko, “A Splittable DNN-Based Object Detector for Edge-Cloud Collaborative Real-Time Video Inference,” in Proc. IEEE AVSS, pp. 1 - 8, 2021.

[5] P. Amanatidis, D. Karampatzakis, G. Iosifidis, T. Lagkas, A. Nikitas, “Cooperative Task Execution for Object Detection in Edge Computing: An IoT Application,” Applied Sciences, vol. 13, no. 8, Art. 4982, 2023.