게임이론 기반 Adaptive Partial Offloading을 적용한 이기종 센서 융합 보행자 안전 시스템

양시훈

가천대학교

tlgnssihun@gachon.ac.kr

A Game-Theoretic Adaptive Partial Offloading Approach for Pedestrian Safety with Heterogeneous Sensor Fusion

Yang Si Hun

Gachon Univ.

요 약

본 논문은 자원 상태에 따라 실시간으로 연산 위치를 결정하는 Adaptive Partial Offloading 기법과 이를 최적화하는 게임이론 기반 오프로딩 결정 모델을 제안한다. 제안 시스템은 CPU 사용률, 배터리 잔량, 네트워크 지연 등 기기의 메트릭을 기반으로 총 네 가지 처리 조합(Case 1~4) 중 센서별 최적 연산 위치(로컬 또는 서버)를 동적으로 선택한다. 음향 데이터는 스마트폰에서 TF Lite 모델로 처리하거나 서버로 오프로딩되며, 레이더 데이터는 ESP32에서 추정된 velocity와 distance 값을 기반으로 서버에 전송된다. 서버는 Sliding Window 기반 타임스탬프 정렬 및 DNN 융합 모델을 통해 데이터를 통합 분석하고 보행자 안전을 위한 위험도를 추론한다. 이기종 센서에 대한 유연하고 통합적인 처리를 통해, 네트워크 품질 저하나 단말 자원 부족 상황에서도 높은 안정성과 실시간성을 유지한다. 실제 환경에서의 실험 결과, 평균 32ms의 낮은 지연 시간, 96.5%의 감지 정확도, 최대 18%의 배터리 효율 향상을 달성하였다. 본 연구는 엣지 기반 센서 융합 시스템에서의 지능형 오프로딩 전략으로서 높은 실효성과 확장 가능성을 입증한다.

Ⅰ. 서 론

1.1 연구 배경

기존 오프로딩 연구는 단일 센서나 단일 디바이스를 기준으로 설계되어, 센서 특성이 상이한 실제 IoT 환경에서는 적용에 한계가 있다. 특히 기존 오프로딩 프레임워크는 동일한 연산 특성을 가정하므로, 이기종 센서 환경에서의 적용 가능성에 제약이 존재한다. 본 연구는 자원 상태에 따라 센서별 연산 위치를 유동적으로 조정하는 Adaptive Partial Offloading 기법을 제안하며, 이를 통해 실시간성과 안정성을 동시에 확보하는 이기종 센서 융합 시스템을 구현한다.

1.2. 연구 목적

본 연구는 다음의 목표를 가진다:

1) 이기종 센서별 연산 및 상황에 따라 로컬·서버 연산을 유연하게 전환하는 Adaptive Partial Offloading 프레임워크를 구현한다.

2) 단말의 자원 상태를 반영하여 센서별 전략을 동적으로 결정하는 게임이론 기반 Offloading Decision Module을 설계한다.

3) 클라우드 Edge 역할의 서버에서 Sliding Window 기반 센서 동기화 및 DNN 융합 추론을 통해 위험도를 실시간으로 정확하게 판단한다.

Ⅱ. 시스템 구성

기존 MEC 아키텍처의 오프로딩 방식들은 전체 연산을 서버로 위임하거나 단일 디바이스 기준으로 설계돼, 센서 특성이 상이한 이기종 환경에서는 한계가 있다. 본 연구는 센서별 연산 비용과 상황을 고려해, 로컬 또는 서버 처리를 동적으로 결정하는 Adaptive Partial Offloading 프레임워크를 제안한다. 이를 통해 상황에 따라 로컬 처리, 서버 오프로딩, 또는 혼합 전략을 유연하게 적용할 수 있다.

2.1. 시스템 구성 및 데이터 흐름

본 시스템은 다음 세 가지 컴포넌트로 구성된다:

Android 앱	음향 데이터 수집 및 추론
ESP32 디바이스	mmWave 레이더 데이터 전송
AWS EC2 서버	Spring Boot 통합 서버 및 FastAPI 추론 서버

이들은 WebSocket 또는 REST API를 통해 통신하며, millisecond 단위의 timestamp를 기준으로 데이터가 동기화된다.

2.1.1 음향 데이터 처리

Android는 44.1kHz PCM 오디오를 수집하고, TFLite 모델을 통해 차량 접근 확률(sound_detected)을 추론한 뒤 WebSocket으로 서버에 전송한다. 오프로딩 모드에서는 PCM을 FastAPI로 전송하여 서버 측에서 동일한 모델로 추론을 수행한다.

2.1.2 레이더 데이터 처리

ESP32는 TRM-121A 센서를 통해 추출한 velocity, distance 값을 WebSocket으로 전송한다. 로컬 추론 시에는 radar_detected score만 전송하며, 오프로딩 시에는 velocity와 distance를 그대로 서버로 전송하여 추론에 사용한다.

2.1.3 서버 측 데이터 처리

Spring Boot 서버는 WebSocket을 통해 실시간 데이터를 수신하며, ±400ms 이내의 데이터를 SlidingWindowMatcher로 동기화한다. 센서 조합에 따라 Case1~4로 분기되고 :

- Case1: 두 센서 모두 로컬 연산 → 서버에서 룰 기반 위험도 평가
- Case2~4: 하나 이상 오프로딩 → FastAPI /predict 호출로 AI 추론

최종 위험도(LOW, MEDIUM, HIGH)는 Android에 WebSocket으로 전송되며, 시스템은 센서 연산 흐름이 유실되지 않도록 구성되었다.

Ⅲ. 시스템 구현

본 시스템은 센서별 자원 상태를 반영하여 로컬 연산과 오프로딩을 유연하게 선택하는 Adaptive Partial Offloading 구조를 채택한다. 서버는 Android 및 ESP32로부터 수집한 메트릭(CPU 사용률, 배터리 잔량, RTT, 대역폭 등)을 기반으로, 아래 4가지 Case 중 최적의 조합을 선택하여 클라이언트에 전달한다. 클라이언트는 전달받은 모드에 따라 연산 경로를 실시간 전환하며 동작한다.

3.1 오프로딩 조합 정의

Case	Sound 처리 위치	Radar 처리 위치
1	Local (Android)	Local (ESP32)
2	Offload (Server)	Local (ESP32)
3	Local (Android)	Offload (Server)
4	Offload (Server)	Offload (Server)

3.2 오프로딩 결정 알고리즘

서버는 클라이언트(Android, ESP32)로부터 수신한 다양한 메트릭(timestamp, battery level, CPU 사용률, RTT, 대역폭)을 기반으로 각 센서(Sound, Radar)에 대해 로컬 연산과 오프로딩의 예상 비용을 비교하여 최적의 처리 위치를 결정한다.비용 함수는 다음과 같이 정의된다:

$$C_{\text{local}} = \alpha E_{\text{local}} + \beta L_{\text{local}}, \quad C_{\text{offload}} = \alpha E_{\text{tx}} + \beta L_{\text{tx}} \quad \cdots \ (1)$$

여기서 E는 에너지 소비, L은 지연 시간, α=0.6, β=0.4는 가중치이다. 서버는 이 두 값을 비교하여 더 낮은 비용을 갖는 방식을 선택하고, 이에 따라 센서별 처리 위치를 정한 후 해당 Case(1~4)를 클라이언트에 전송한다. 예를 들어, 로컬 연산은 지연이 적은 대신 배터리 소모가 크며, 오프로딩은 에너지 측면에서 유리하지만 네트워크 지연에 민감하다.

각 클라이언트는 전달받은 모드에 따라 연산 경로를 동적으로 전환한다. Android는 Case 2·4의 경우 PCM 오디오를 FastAPI 서버로 업로드하고, Case 1·3에서는 로컬에서 추론한 결과(sound_detected)를 전송한다. ESP32는 velocity, distance 또는 radar_detected 값을 WebSocket으로 전송하며, 이 역시 모드에 따라 결정된다.

이렇게 해당 시스템은 각 센서의 상태를 기반으로 비용을 계산하고, 4가지 오프로딩 조합 중 최적의 전략을 선택하는 방식으로, 실시간성 유지와 에너지 효율을 동시에 확보할 수 있다.

Ⅳ. 실험 및 성능 평가

본 시스템은 Android 단말, ESP32-CAM, TRM-121A mmWave 레이더, AWS EC2 기반 서버로 구성되었으며, Wi-Fi 6 환경에서 Adaptive Offloading의 성능을 검증하였다. Android와 ESP32는 메트릭 정보를 서버에 주기적으로 전송하고, 서버는 이를 기반으로 최적의 오프로딩 Case를 선택하여 클라이언트에 전달한다. 클라이언트는 이에 따라 연산 경로를 전환하고, 센서 데이터는 서버에서 Sliding Window 기반으로 병합·동기화되어 DNN 추론에 활용된다.

· 레이턴시: Adaptive 모드는 자원 상태에 따라 Case를 전환하며 평균 32ms의 최저 지연 시간을 기록하였다. (Figure 1)

· 정확도: 센서 동기화 및 DNN 융합 추론을 통해 F1 Score 96.5%의 감지 정확도를 달성하였다. (Figure 2)

· 배터리 효율: Adaptive 방식은 Case 4 대비 최대 18%의 배터리 절감 효과를 보였다.(Figure 3)

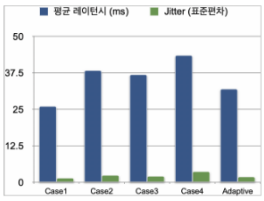

Figure 1 : 레이턴시 비교

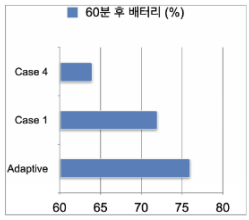

Figure 2 : 배터리 비교

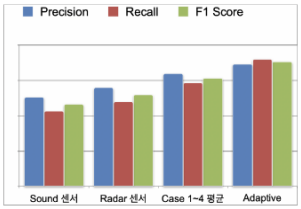

Figure 2 : 정확도 비교

Ⅴ. 결론 및 향후 과제

4.1 추가 연구

기존 Adaptive Partial Offloading은 고정된 비용 모델을 기반으로 연산 위치를 결정하지만, 향후에는 보다 정교한 판단이 요구된다. 이를 위해 각 센서를 자율 에이전트로 간주하고, 게임이론 기반의 오프로딩 판단 전략을 도입할 수 있다.

$$U_i = R_i - \alpha E_i - \beta L_i \cdot \cdots \ (2)$$

여기서 U_i는 센서 i의 유틸리티, R_i는 오프로딩을 통해 기대되는 정확도 향상을 의미하며, E_i, L_i는 각각 에너지소비와 레이턴시를 의미한다. 이 모델은 정확도 향상, 에너지 소비, 지연 시간 간의 trade-off를 반영하며, payoff가 임계값을 초과할 경우에만 오프로딩을 수행하는 구조로 확장 가능하다. 향후 다중 사용자 환경을 고려한 분산 전략 모델링도 가능하다.

4.2 결론

본 연구는 mmWave 레이더와 음향 센서를 융합한 보행자 안전 시스템에 Adaptive Partial Offloading을 적용하여, 센서별 리소 상태에 따라 연산 위치를 실시간 전환하는 구조를 구현하였다. 서버는 수집된 메트릭을 기반으로 최적 Case를 결정하고, Sliding Window 동기화와 DNN 추론을 통해 위험도를 평가한다.

실험 결과, 평균 32ms 지연, 96.5% 감지 정확도, 최대 18% 배터리 절감을 달성하며 시스템의 효과를 입증하였다. 본 연구는 이기종 센서 환경에서의 실시간 오프로딩 전략 설계의 실질적 가능성을 제시한다.

ACKNOWLEDGMENT

This research was supported by the Wireless Intelligence Networking Lab (WINL) at Gachon University.

참 고 문 헌

[1] S. Xia, et al., "Improving Pedestrian Safety Using Intelligent Systems," IEEE Internet of Things Journal, 2019.

[2] L. Zhao, X. Fei, and Q. Wang, "Modeling Task Offloading in Mobile Edge Computing: A Game Theoretic Approach," Proceedings of the 8th International Conference on Computer and Data Engineering (ICCDE), 2022.