딥러닝을 이용한 엣지기반 실시간 교통 통행량 산출 방법

이준구, 백장운, 임길택
한국전자통신연구원
{leejg01679, jwbaek98, ktl}@etri.re.kr

Edge-based Real-time Method of Counting Traffic Volume Using Deep Learning

Joon-Goo Lee, Jang Woon Baek, Kil-Taek Lim
Daegu-Gyeongbuk Research Center
Electronics and Telecommunications Research Institute

요 약

본 논문은 딥러닝을 이용한 엣지 기반 실시간 교통 통행량 산출 방법을 제안한다. 제안 방법은 딥러닝 알고리즘을 교통 영상에 적용하여 교통 객체를 검출 및 추적하고, 추적된 궤적 정보를 이용한 방향별 통행량을 산출한다. 교통 객체 검출기는 엣지 단말에서 실시간으로 동작할 수 있도록 경량 네트워크를 사용하였으며, 특정 사이트에 적응할 수 있도록 재학습을 통해 성능 향상을 도모하였다. 제안하는 방법은 엣지 단말에서 96% 이상의 계수 정확도와 90FPS 이상의 처리 속도를 보였다.

Ⅰ. 서 론

교통 통행량 정보는 교통과 관련된 시간 및 비용을 효과적으로 운용하기 위한 필수 정보이다. 기존의 통행량 계수는 녹화 영상에 대해 인력을 이용하거나 루프 센서 검지기를 주로 사용하였다. 인력을 이용하는 방법은 단순 반복적인 작업으로 인해 높은 비용이 소요되며, 검지기를 이용하는 방법은 설치비에 비용이 많이 들고 유지 보수가 어려운 단점이 있다.

현재 다양한 분야에 딥러닝 기술을 적용하여 교통 객체를 검출하는 방법들이 많이 소개되고 있으며, 특히 VGG[1], ResNet[2] 등의 우수한 백본 네트워크들과 Faster-CNN[3], SSD[4], YOLO[5] 등의 검출기들이 널리 사용되고 있다. 하지만 이러한 네트워크들은 방대한 클래스와 다양한 환경에 범용적으로 사용되기 위한 목적으로 설계되어 고성능 서버 환경에 적합하고 엣지 환경의 실시간 교통 객체 검출에는 적합하지 않다.

본 논문에서는 딥러닝 기술을 교통 감시 영상에 적용하여 실시간으로 교통 객체를 검출 및 추적하고, 추적된 궤적 정보를 이용하여 방향별 통행량을 산출하는 방법을 제안한다.

Ⅱ. 본론

본 논문에서는 경량 딥러닝 알고리즘 기반의 실시간 통행량 산출 방법을 제안한다. 제안 시스템의 구조는 그림 1과 같이 서버와 엣지 단말로 구분한다. 서버에서는 AICity, AVSS, MIO-TCD, CNRPark 등의 교통 객체와 관련된 OpenDB를 이용하여 범용 검출기를 학습한다. 여기서 사용되는 검출기는 자원이 한정된 엣지 단말에서 실시간으로 작동하기 위해 입력 영상 사이즈, 네트워크 레이어 및 필터의 수를 줄이는 등의 경량화를 수행한다. 경량화된 범용 검출기를 다양한 환경의 엣지 단말에서 그대로 사용할 경우 충분한 검출 성능을 보장하지 못한다. 따라서 성능 최적화를 위해 사이트 별로 영상을 다운로드 하여 학습데이터를 생성한 후 범용 검출기를 초기 가중치로 하여 엣지 단말별 검출기를 각각 학습한다.

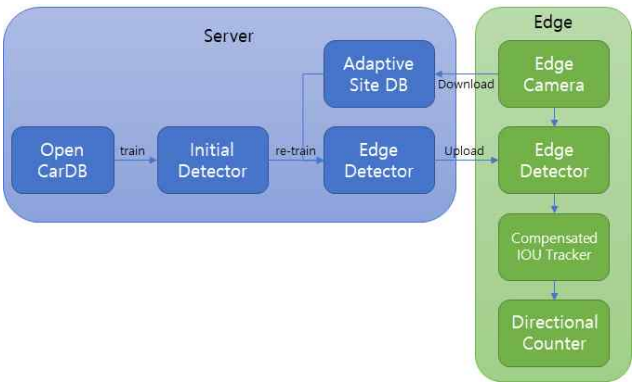

그림 1. 교통 통행량 산출 방법의 구조

엣지 단말에서는 서버에서 최적화되어 업로드된 검출기를 이용하여 입력 영상에서 교통 객체를 실시간으로 검출한다. 추적기는 검출된 객체 정보를 기반으로 시간적으로 동일한 객체의 움직임을 추적하며, 각 객체별 이동 궤적을 결과로 출력한다. 여기서 추적기는 엣지 단말에서 실시간으로 작동하기 위해 IOU 기반의 추적기를 사용하며, 오검출 및 미검출 등으로 발생하는 오류를 방지하기 위해 박스 크기의 변화량, 중심 이동 거리, 중심 이동 방향, 평균 컬러 등의 정보를 이용해 보정한다. 추적이 완료된 객체로부터 출력되는 이동 궤적을 이용하여 각 교통 객체의 수와 진출 방향을 구분하여 방향별 계수 정보를 출력한다.

실험을 위해 그림 2와 같이 특정 사거리의 주간, 야간, 비 등의 다양한 환경에서 12가지의 진출입이 포함된 8,000 프레임을 추출하였으며, 훈련된 전문 인력을 이용하여 검출, 추적, 계수 등의 정보를 태깅하였다. 실험에 사용된 서버는 Intel i9-9900K 3.60GHz CPU, RTX-2080Ti GPU를 사용하였다.

a. 저녁 b. 야간

c. 비 d. 맑음

그림 2. 테스트 영상

표 1은 서버에서 YOLOv3-SPP를 사용했을 때, 검출, 추적, 계수의 성능을 나타낸다. 계수 정확도는 약 96% 이상으로 우수한 성능을 보였지만 서버급의 사양에서 26ms의 검출 속도는 엣지 보드에서 실시간 처리를 제공하지 못한다. 이러한 문제를 해결하기 위해 검출 성능을 보장하면서 실시간성을 만족하기 위해 속도가 빠른 경량화 된 네트워크를 설계하였다.

표 1. YOLOv3-SPP 기반 성능 테스트

YOLOv3-SPP	Time	Performance
Detection(AP)	26ms	95.00%
Tracking(MOTP[6])	1ms	96.99%
Counting(Precision)	1ms	96.00%

표 2는 YOLO 기반 기존의 두 가지 네트워크와 제안하는 경량 네트워크를 GPU 기반과 CPU 기반의 엣지 보드에 적용한 각각의 결과를 비교하여 보여준다. YOLOv4는 기존의 YOLO보다 높은 검출률을 보이며, YOLOv3-tiny는 YOLOv3를 경량화하여 처리 속도를 높였다. 제안하는 알고리즘은 YOLOv4의 검출률을 유지하면서 YOLOv3-tiny보다 빠른 처리속도를 위해 네트워크를 개선하였다. YOLOv4와 비교하여 검출률은 0.7% 낮으나 최대 약 4배 이상의 처리율을 보였으며, YOLOv3-tiny와 비교하여 검출률은 5.3% 높으면서 약 1.5배 이상의 처리율을 보였다. 특히 경량 네트워크는 엣지 기반의 GPU보다 CPU에서 약 1.5배 이상의 처리 성능을 보였으며, 이는 GPU 사용에 따른 오버헤드로 보인다.

표 2. 엣지 보드 기반 속도 비교 테스트

		YOLOv4	YOLOv3-tiny	Ours
AP		97.20%	91.20%	96.50%
Bflops		35.244	3.405	2.356
layers		162	28	36
weights		244MB	18.4MB	4.4MB
FPS	RTX2080Ti	78.4	106.1	106.7
	XavierNX	14	57.6	61.2
	i7-8665UE	unsupported	66.94	96.56
	i5-8365UE	unsupported	54.38	85.01
	i3-8145UE	unsupported	42.25	60.01

그림 3. 통행량 계수 결과

Ⅲ. 결론

본 논문에서는 딥러닝을 이용한 엣지 기반의 통행량 계수 방법을 제안하였다. YOLO 기반의 검출기를 사용하여 차량을 검출하고, Custom IOU 트랙커를 이용하여 차량을 추적하였으며, 추적된 궤적을 이용해 방향별 통행량을 계수하였다. 또한, 엣지 환경에 적용하기 위해 검출기 네트워크를 경량화하여 약 96% 이상의 높은 검출률과 약 96FPS 이상의 처리 속도로 개선하였다. 검출, 추적, 계수 결과는 그림 3과 같이 직관적 인터페이스로 표현하여 실제 환경에 적용 가능성을 보였다.

ACKNOWLEDGMENT

This work was supported by Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean government. [20ZD1100, Development of ICT Convergence Technology for Daegu-GyeongBuk Regional Industry]

참 고 문 헌

[1] Simonyan. Karen, and Andrew Zisseman, " Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1566, 2014.

[2] He K., Zhang X. Ren S. and Sun J., "Deep residual learning for image recognition, " *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 770-778, 2016.

[3] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *arXiv:1506.01495 [cs.CV]*.

[4] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu, and A. Berg, "SSD: Single Shot MultiBox Detector," *arXiv:1512.02325 [cs.CV]*.

[5] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv:1804.02767 [cs.CV]*.

[6] Bernardin K., Elbs A., and Stiefelhagen R., "Multiple object tracking performance metrics and evaluation in a smart room environment," Sixth IEEE International Worksop on Visual Surveillance in conjunction with ECCV, Vol.90, p. 91, 2006.