실시간 컴퓨터 비전 처리를 위한 영상 전송 파이프라인 시스템 연구

이재성, 이현민, 김민수, 심효은, 김정석*, 고석주
경북대학교, * SK Telecom

lee01042000@gmail.com, hymi54@knu.ac.kr, minsue9608@knu.ac.kr ,

a2921641@knu.ac.kr, jeongseok.kim@sk.com*, shilla@slu.ac.kr

A Study on the video streaming pipeline system for real-time computer vision processing

Lee Jaeseong, Lee hyunmin, Kim Minsu, Sim Hyoeun, Kim Jeongseok *, Go sukju
Kyungpook National Univ., *SK Telecom

요 약

머신 러닝 기술은 다양한 산업에 적용되고 있다. 그 중 컴퓨터 비전 모델의 경우 성능이 높은 모델은 고수준의 연산능력을 요구한다. 따라서 컴퓨터 비전 모델을 IoT 분야에 적용할 경우 제한된 컴퓨팅 환경 때문에 라즈베리파이 4 같은 에지 디바이스에서 활용하기는 어렵다. 본 논문에서는 이를 극복하기 위해서 영상 전송 파이프라인을 이용하여 이미지를 서버로 전송, 처리하여 결과를 수신하는 시스템을 제안한다.

I. 서 론

최근 머신 러닝 기술이 발전하면서 다양한 산업 분야에 컴퓨터 비전을 활용하는 연구 및 개발이 이루어지고 있다. 컴퓨터 비전은 인간의 시각으로 할 수 있는 일을 수행하는 자율적인 시스템을 만드는 것을 목표로 한다. 컴퓨터 비전 기술이 실용화가 되기 위해서는 컴퓨터 비전 기술을 탑재한 시스템의 안정성과 정확성이 중요하다. 이를 위해서는 높은 퍼포먼스를 가진 학습 모델을 사용해야 한다. 높은 퍼포먼스를 가진 학습 모델은 고수준의 연산 능력이 갖춰진 환경을 요구한다. 그러나 IoT 같은 에지 디바이스 에서 컴퓨터 비전 기술을 활용할 때, 100ms 이내로 Just-In-Time(JIT) 실행이 어렵다.[1] IoT 산업에서 기술을 실용화하기 위해서는 제한된 컴퓨팅 환경에서 높은 퍼포먼스의 학습 모델을 사용하도록 하는 방법이 필요하다. 본 논문에서는 위 방법을 위해서 영상 전송 파이프라인을 개발하고 서버에서 컴퓨터 비전을 처리하여 에지 디바이스에서 높은 퍼포먼스의 학습 모델을 사용할 수 있는 방안을 제시한다.

II. 관련 연구

에지 디바이스에서 머신 러닝 기술을 좀 더 효율적으로 활용할 수 있도록 구글에서 TensorFlow Lite[2]을 공개했다. TensorFlow Lite 는 TF Lite interpreter, TF Lite converter 이 두 가지 주요 컴포넌트로 구성되어 있는데 TF Lite interpreter 는 최적화된 모델을 다양한 하드웨어에서 돌아갈 수 있도록 하고, TF Lite converter 는 모델을 인터프리터가 사용할 수 있는 효율적인 형태로 바꿔주는 최적화 기능을 제공한다. 하지만 라즈베리파이 4 와 같은 CPU 만 존재하는 에지 디바이스의 경우 최적화를 진행해도 JIT 실행이 어렵다. 소규모의 GPU 가 있는 에지 디바이스의 경우 또한 중간 성능의 컴퓨터 비전 모델을 실행 시킬 수 있지만 고성능의 컴퓨터 비전 모델을 JIT 실행시키는 것은 어렵다. 아래 표 1 은 에지 디바이스와 GPU 의 성능을 보여주고 있다.

표 1. Edge Device, CPU 성능 비교[3]

Edge Device	MobileNet v2, ms	ResNet-50 , ms	VGG-19 , ms
NVIDIA Jetson Nano	15.652	27.78	100
Raspberry Pi 3	400	714.29	2000
Raspberry Pi 3 + Intel Neural Compute	33.33	62.5	200
GPU	**MobileNet v2, ms**	**Inc-ResNet v1, ms**	**VGG-19 , ms**
GeForce GTX 1080 Ti	1.5	9.5	10
GeForce GTX 950	3.9	38	47
NVIDIA Tesla K40c	3.7	38	60

표 1 에서 MobileNetV2 와 같은 소형 모델에서는 JIT 실행이 가능하지만 성능이 낮다는 단점이 있다. VGG-19 모델의 경우는 100ms 이상으로 JIT 실행이 불가능한 것을 볼 수 있다. 그러나 GPU 의 경우 모두 JIT 실행이 가능한 것을 볼 수 있다.

본 논문에서는 에지 디바이스에서 모델을 수행하는 것을 최적화하는 방향이 아니라 필요한 연산을 고성능의 연산능력이 있는 서버에서 대신 수행하여 에지 디바이스의 한계를 극복하는 연구를 진행한다.

III. 수행 환경

본 논문에서는 실험을 위해 기기에서 얻은 이미지를 GStreamer[4] 라이브러리를 이용하여 실시간으로 서버로

전송해서 서버에서 모델을 수행하고 수행한 결과를 다시 기기로 전송하는 파이프라인을 개발했다.

통신의 경우 통제 불가능한 요소를 제거하고 지연율과 대역폭에 따른 변화를 관찰하기 위해 랜 케이블을 이용해서 직접적으로 연결하여 0ms 에 가까운 레이턴시 환경을 구성하고 네트워크 에뮬레이터를 이용하여 유사 환경을 구축하였다. 네트워크 에뮬레이터 설정 값의 경우 표 2 의 값을 사용하였다.

표 2. 통신 측정 결과 (MB/s, 부산광역시 사하구 일대)

		Download	Upload	Bandwidth	Latency(ms)
4G	AVG	12.1254	2.8718	97.021	28.265
	MIN	2.59	0.79	20.7	24
	MAX	31.7	5.73	254	37.8
	STDEVS	7.95	1.49	63.70	3.16
5G	AVG	92.156	7.7577	737.46	18.773
	MIN	48.4	2.86	387	16.3
	MAX	123	12.6	985	96.5
	STDEVS	17.36	2.10	139.07	8.18

표 2 의 설정값은 부산 광역시 사하구 일대에서 SM-G981N(갤럭시 S20)을 이용하여 실제 4G 와 5G 를 이용한 경우에 측정된 평균 지연율과 평균 대역폭 값을 사용하였다.

모델의 경우 컴퓨터 비전 중에서도 객체 탐지 모델을 선택하였다. Pytorch Framework 에서 실행하였으며 Torchvision.mode[5] 라이브러리에 있는 4 개의 객체 탐지 모델을 사용하였다. 각 모델은 실행할 서버 GPU 는 GTX1060 과 RTX3080 을 사용하였다. 표 3 는 사용된 모델명과 각각의 성능이다.

표 3. COCO 사전 학습된 객체 탐지 모델 성능

Network	Box AP
Faster R-CNN ResNet-50 FPN	37.0
Faster R-CNN MobileNetV3-Large FPN	32.8
Faster R-CNN MobileNetV3-Large 320 FPN	22.8
RetinaNet ResNet-50 FPN	36.4

표 3 의 성능은 COCO 데이터셋으로 훈련된 모델을 테스트하여 측정한 정확도 결과이다.

IV. 수행 결과

라즈베리파이 4, GTX1060 그리고 RTX3080 에서 224x224 크기를 가진 50 장의 이미지를 추론하도록 하여 평균 추론 시간을 측정하였다. 표 4 는 각 모델을 적용한 측정결과이다.

표 4. 이미지 평균 추론 시간

Network	Raspberry Pi4	GTX 1060	RTX 3080
Faster R-CNN ResNet-50 FPN	3,356.82ms	14.88ms	3.37ms
Faster R-CNN MobileNetV3-Large FPN	992.38ms	5.16ms	1.63ms
Faster R-CNN MobileNetV3-Large 320 FPN	387.38ms	3.70ms	1.47ms
RetinaNet ResNet-50 FPN	3855.07ms	15.54ms	3.40ms

측정 결과 라즈베리파이 4 에서는 실용적인 사용이 불가능할 정도로 높은 추론 시간이 측정되었으며 그래픽카드의 경우 20ms 이내의 매우 낮은 추론 시간이 측정되었다.

표 5. 통신 포함 객체 탐지 모델 실험 지연율 결과

	Network	GTX 1060	RTX 3080
4G	Faster R-CNN ResNet-50 FPN	53.67ms	37.53ms
	Faster R-CNN MobileNetV3-Large FPN	38.63ms	34.75ms
	Faster R-CNN MobileNetV3-Large 320 FPN	36.11ms	34.63ms
	RetinaNet ResNet-50 FPN	50.35ms	39.23ms
5G	Faster R-CNN ResNet-50 FPN	49.73ms	33.36ms
	Faster R-CNN MobileNetV3-Large FPN	34.32ms	29.81ms
	Faster R-CNN MobileNetV3-Large 320 FPN	32.98ms	29.72ms
	RetinaNet ResNet-50 FPN	46.38ms	34.18ms

우선 네트워크 에뮬레이터를 이용하여 가상의 4G 와 5G 환경을 구성하여 각각 실험하였다. 이후 라즈베리파이 4 에서 카메라로 이미지를 생성한 다음 224x224 의 크기로 사이즈를 조절하고 프로토콜을 사용하여 그래픽카드가 있는 서버로 전송하였다. 서버는 수신한 영상을 프레임 단위로 추론하도록 하고 객체 탐지 수행의 결과로 Bounding Box 의 범위와 확률 값을 출력하게 된다. 이후 Bounding Box 의 정보를 다시 라즈베리파이 4 로 SRT 프로토콜[6]을 사용하여 전송하도록 하였다. 라즈베리파이 4 는 수신한 정보를 원본 이미지에 합성하여 객체 탐지의 결과 영상을 출력하도록 한다. 위 지연율은 카메라에서 이미지를 생성한 직후와 객체 탐지의 결과 영상을 출력한 직후의 시차를 계산하고 이를 30 회 반복한 평균값이다.

V. 결론

수행 결과 라즈베리파이 4 에서 모델을 실행한 경우 실용화를 위한 JIT 추론이 불가능했지만 본 논문에서 제안한 파이프라인을 활용한 결과 지연율을 50ms 이내로 낮추어 이를 극복하였다. 또한 5G 환경 뿐만 아니라 4G 환경에서도 JIT 이내의 결과를 달성하였다는 것 을 볼 수 있다. 그리고 GTX1060 이 RTX3080 보다 평균 5 배의 추론 시간을 가지고 있지만 라즈베리파이 4 에 비하면 매우 짧아 에지 디바이스보다 상대적으로 높은 추론 연산 능력을 가진 그래픽카드를 사용해면 준수한 결과를 가질 수 있다는 점을 시사하고 있다. 향후 실제 통신환경에 적용하여 유효성을 검증할 예정이다.

ACKNOWLEDGMENT

본 연구는 과학기술정보통신부 및 정보통신기획평가원의

SW 중심대학지원사업 (2021-0-01082)으로 수행되었음

참 고 문 헌

[1] Inference performance results from Jetson Nano, Raspberry Pi 3, Intel Neural Compute Stick 2, and Google Edge TPU Coral Dev Board, Jetson Nano: Deep Learning Inference Benchmarks, *Nvidia Developer*, https://developer.nvidia.com/embedded/jetson-nano-dl-inference-benchmarks

[2] TensorFlow Lite for Mobile & IoT tutorials, guide, example, API, https://www.tensorflow.org/lite/guide?hl=ko

[3] Machine Learning Performance Result, AI-Benchmark, https://ai-benchmark.com/ranking.html

[4] GStreamer : open source multimedia framework, https://gstreamer.freedesktop.org/

[5] Object Detection, Instance Segmentation and Person Keypoint Detection, TORCHVISION.MODELS, Pytorch Docs, https://pytorch.org/vision/stable/models.html#torchvision-models

[6] Jeongseok Kim, Jaeho Lee, *High Quality Video Streaming System in Ultra-Low Latency over 5G-MEC*, 2021 https://doi.org/10.3745/KTCCS.2021.10.2.2