KIPS Trans. Softw. and Data Eng.
Vol.9, No.10 pp.323~328
ISSN: 2287-5905 (Print), ISSN: 2734-0503 (Online)

https://doi.org/10.3745/KTSDE.2020.9.10.323

Real Time Face detection Method Using TensorRT and SSD

Hye-Bin Yoo† · Myeong-Suk Park†† · Sang-Hoon Kim†††

ABSTRACT

Recently, new approaches that significantly improve performance in object detection and recognition using deep learning technology have been proposed quickly. Of the various techniques for object detection, especially facial object detection (Faster R-CNN, R-CNN, YOLO, SSD, etc), SSD is superior in accuracy and speed to other techniques. At the same time, multiple object detection networks are also readily available. In this paper, among object detection networks, Mobilenet v2 network is used, models combined with SSDs are trained, and methods for detecting objects at a rate of four times or more than conventional performance are proposed using TensorRT engine, and the performance is verified through experiments. Facial object detector was created as an application to verify the performance of the proposed method, and its behavior and performance were tested in various situations.

Keywords : Tensorflow, TensorRT, Deep Learning, SSD, Object Detection

TensorRT와 SSD를 이용한 실시간 얼굴 검출방법

유 혜 빈† · 박 명 숙†† · 김 상 훈†††

요 약

최근에는 딥러닝 기술을 이용하여 물체 검출 및 인식에서 성능이 크게 향상되는 새로운 접근방법들이 빠르게 제안되고 있다. 객체, 특히 얼굴객체 검출에 관한 여러 기법(Faster R-CNN, R-CNN, YOLO, SSD 등) 중 SSD는 다른 기법들보다 정확도와 속도에서 우수하다. 동시에 여러 객체 검출 네트워크들(object detection network)도 쉽게 이용할 수 있다. 본 논문에서는 객체 검출 네트워크 중 Mobilenet v2 network를 이용하고 SSD와 결합한 모델을 훈련하고, TensorRT engine을 이용하여 기존의 성능보다 4배 이상의 속도로 객체를 검출하는 방법에 대해 제안하고 실험을 통해 성능을 검증한다. 제안한 방법의 성능 검증을 위한 응용으로 얼굴객체 검출기(facial object detector)를 만들어 다양한 상황에서 동작과 성능을 실험하였다.

키워드 : 텐서플로우, 텐서알티, 딥러닝, 에스에스디, 객체 검출

1. 서 론

객체 검출(Object Detection)이란 목표 영상 내에서 시각적으로 관심이 있는 영역의 위치를 지정하고(Localization), 지정 위치에 존재하는 관심 영역의 범위를 제한하며(Area Detection), 관심 영역 내의 객체가 어떤 종류인지를 분류(Classification)하는 과정이다. 2012년 이후, AlexNet [4]의 등장으로 객체 검출에 관한 문제를 딥러닝(Deep-learning)을 활용한 성능 개선에 접근하기 시작하였다.

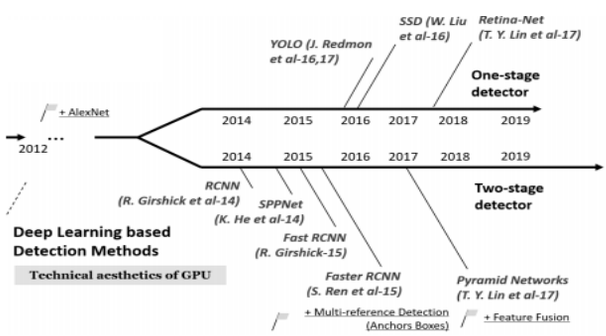

Fig. 1. Object Detection Technology with Deep-learning [1]

2012년에 Image Classification 대회 ILSVRC(ImageNet Large Scale Visual Recognition Challenge)에서 AlexNet이라는 네트워크가 등장하게 된다. 이 대회를 기점으로 딥러닝을 활용하여 문제를 풀기 시작하였다. 즉, Fig. 2에서 처럼

※ 본 연구는 2019학년도 한경대학교 연구년 경비의 지원에 의한 것임.
※ 이 논문은 2020년 한국정보처리학회 춘계학술발표대회의 우수논문으로 "TensorRT 엔진과 SSD를 이용한 Face detection"의 제목으로 발표된 논문을 확장한 것임.
† 준 회 원 : 한경대학교 ICT로봇기계공학부 학사과정
†† 준 회 원 : 한경대학교 ICT로봇기계공학부 박사수료
††† 종신회원 : 한경대학교 ICT로봇기계공학부 교수
Manuscript Received : July 7, 2020
First Revision : August 20, 2020
Accepted : August 26, 2020
* Corresponding Author : Sang-Hoon Kim(kimsh@hknu.ac.kr)

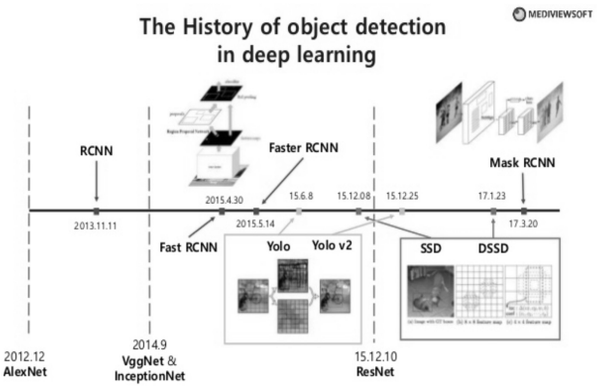

Fig. 2. Changing Object Detection Techniques Using Deep-learning [1]

CNN과 같은 이미지에 특화된 딥러닝 방법을 이용하여 이러한 문제들을 풀기 시작한 것이다.

딥러닝을 사용하는 문제로는 크게 2단계 기반 검출기(Two-Stage Detectors)와 1단계 기반 검출기(One-Stage Detectors) 두 가지 방법 [1]이 있다.

2단계 기반 검출기란 전체 처리 과정을 2단계에 걸쳐 객체를 검출하는 방법으로, 한 장의 이미지에 매우 많은 네트워크를 사용하기 때문에 엄청난 연산량을 요구하게 된다. 따라서 이 방법으로는 실시간성을 기대하기 어렵다. 이 검출 기법의 개량판도 상황은 같다. Fig. 3은 대표적인 2단계 기반 검출기의 개량판인 Faster R-CNN의 네트워크 모습이다. 이러한 검출기도 최대 처리속도는 7 FPS 정도이다[5]. (with mAP 73.2%)

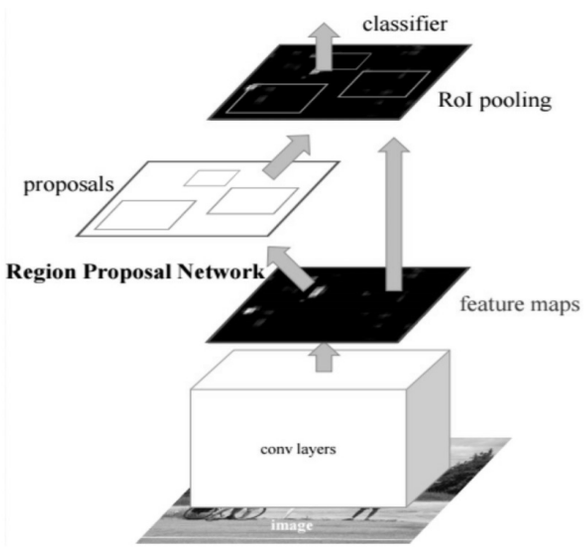

Fig. 3. Faster R-CNN [5] Structure

2015년부터는 1단계 기반 검출방법을 주로 사용하여 더욱 속도가 개선되어 실시간성까지 증명이 되며 객체의 검출(detection)뿐 아니라, 추적(tracking), 분할(segment) 분야까지 널리 활용하게 되었다. Fig. 4에서 보는 것처럼 정확도

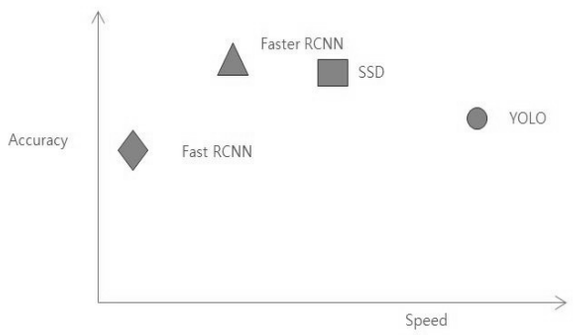

Fig. 4. Deep Learning-Based Vision Algorithm Performance Comparison [6]

는 2단계 기반 검출 방식인 Faster R-CNN이 더 높지만, 속도는 1단계 기반 방식인 SSD와 YOLO가 높다. 이와 같은 비교를 통해 상황에 맞게 알맞은 알고리즘을 선택하여 객체 검출에 사용하는 것이 요구된다.

본 논문에서는 임베디드 환경에서의 빠르고 정확한 실시간 객체 검출을 위하여 1단계 기반 검출 방식의 알고리즘과 적합한 딥러닝 네트워크를 비교한다. 또한, 정확하고 빠른 검출을 위해 딥러닝 인퍼런스 최적화 라이브러리인 TensorRT [9]를 사용하여 그에 따른 속도에서의 성능 향상을 기존의 방법과 비교하고 분석한다.

2. 임베디드환경에 기반한 실시간 객체 검출 기법

2.1 YOLO(You Look Only Once)를 이용한 검출방법

1단계 기반 검출 방식의 알고리즘 중 하나인 YOLO [10]는 네트워크의 최종 출력단에서 경계박스의 위치 찾기와 클래스 분류가 동시에 이뤄진다. 단 하나의 네트워크가 한 번에 특징도 추출하며 경계 상자도 만들고, 클래스도 분류하기 때문에 간단하고 빠른 것이 특징이다. Fig. 5의 좌측의 입력 영상이 네트워크를 통과하면 중앙의 2개의 데이터를 얻는다. 이것이 네트워크의 최종 출력이며 최우 측의 이미지는 네트워크의 최종 출력물을 이용하여 얻어지는 결과이다.

가운데 위쪽은 경계 상자(Bounding Box)에 대한 정보이다. 네트워크는 영상을 7x7 그리드로 나누고, 크기가 일정하지 않은 경계 상자를 2개씩 생성한다. 그리드 셀이 7x7 = 49이므로 경계 상자는 총 49x2 = 98개가 만들어진다. 이 중 경계 상자 안쪽에 어떤 객체가 있을 것 같다는 의미인 컨피던스 점수(confidence score)가 높을수록 상자를 굵게 그려준다. 굵은 경계 상자들만 남기고 얇은 경계 상자를 지운다. 남은 후보 경계 상자들을 NMS(Nom-maximal suppression) 알고리즘을 이용해 선별하면 우측의 이미지처럼 3개만 남게 된다. 가운데 아래쪽은 클래스에 대한 정보이다. 총 49개의 그리드 셀이 만들어지는데 각 그리드 셀은 해당 영역에서 제안한 경계 상자 안의 객체가 어떤 클래스인지 색깔별로 표현하고 있다.

이러한 과정을 거치면 최종적으로 우측의 결과를 얻는다.

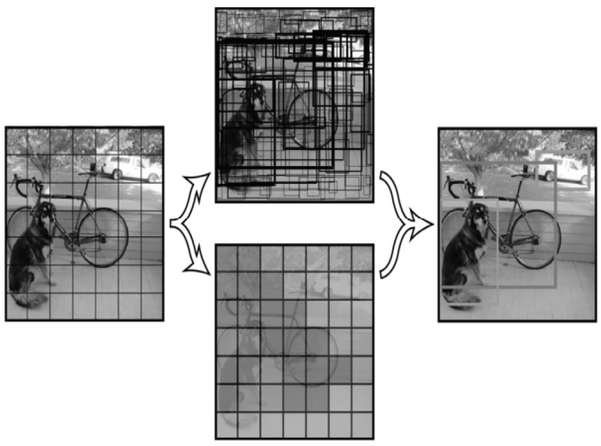

Fig. 5. YOLO [10] Object Detection Process

2.2 SSD(Single Shot Multibox Detector)에 기반을 둔 객체 검출

1단계 기반 검출 방식의 또 다른 알고리즘인 SSD(Single shot multibox detector) [2]는 Single-shot detector라는 표현과 같이 입력 사진의 변형 없이 그 한 장만으로 훈련 및 검출을 하는 검출기를 의미한다. SSD는 후보 영역 추출 과정과 재 표본화(resampling) 과정을 제거한 방식을 이용함으로써 높은 정확성과 빠른 속도를 모두 달성하였다.

하지만 Single-shot learning을 위해서는 한 가지 큰 문제를 해결해야 한다. 단 한 장의 사진만을 가지고 여러 가지 크기의 물체를 검출해야 한다는 것이다. SSD는 이러한 문제를 기본 구조 뒤에 보조 구조를 붙여 얻은 멀티-스케일 특징 맵(Multi-scale feature maps)을 이용하여 해결하였다. Fig. 6 [2]에서 SSD의 구조(architecture)를 보여준다. 기본 구조나 보조 구조에서 얻은 특징 맵(feature map)들은 각각 다른 컨볼루션 필터(convolution filter)에 의해 결과값을 얻게 된다. 또한, 다른 어떠한 검출 모델에 대해서도 Single-shot learning에 이용할 수 있다는 점은 실시간 객체 검출에 관한 연구에 용이하다고 볼 수 있다.

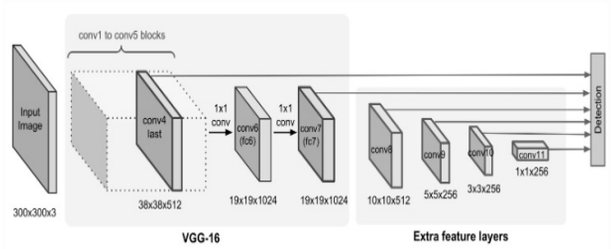

Fig. 6. SSD Architecture [2]

2.3 실시간 객체 검출을 위한 TensorRT의 활용

TensorRT는 Tensorflow에 내장된 TF-TRT와 NVIDIA사에서 개발한 TRT로 크게 두 가지로 나눌 수 있다. 둘은 같은 TensorRT 네트워크 정의를 사용한다는 공통점이 있지만, 그래프를 가져오는 방식에서 크게 차이가 난다. TF-TRT는 Tensorflow 훈련 그래프(Saved model 포맷, frozengraph 포맷)를 가져와 최적화를 진행하며, TRT는 위의 포맷을 UFF나 ONNX의 파서 포맷으로 변환을 한 후 최적화를 진행한다.

위의 두 가지는 최적화하는 범위가 달라서 성능 면에서도 차이를 보인다. TF-TRT는 그래프 일부분에서 최적화가 수행되지만, TRT는 그래프 전체적으로 최적화가 수행되기 때문에 이론적으로 TRT가 더 빠르다. 이와 같은 이유로 본 논문에서 최종적으로 사용할 TensorRT 종류는 TRT이다. 이렇게 성능을 향상하는 것이 가능한 이유는 NVIDIA사의 GPU와 CUDA-X AI등 최신 하드웨어와 기술 등이 결합해 사용되기 때문이다. Fig. 7처럼 TRT 동작은 크게 두 가지로 나뉜다. 첫 단계는 기존 딥러닝 모델을 최적 모델로 변환하는 것이고, 두 번째 단계는 변환된 딥러닝 모델을 TensorRT 실행시간에서 구동시키는 것이다.

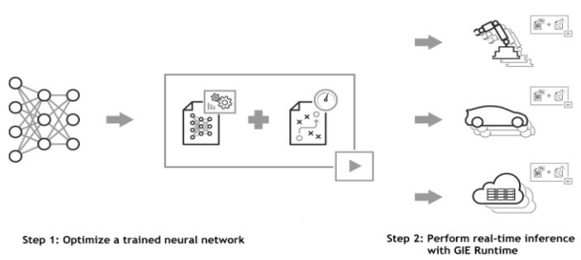

Fig. 7. TensorRT Engine Process [9]

2.4 Mobilenet v2 네트워크의 적용

Mobilenet v1 [7]은 기존의 컨벌류션(Convolution) 연산을 대체하여 DC(Depthwise Convolution)과 PC(Pointwise Convolution)을 혼합하여 사용하는 Depthwise Separable Convolution 방법을 사용한다. Fig. 8과 Fig. 9에서 볼 수 있듯이 DC는 동일 채널 내에서 컨벌류션을 하는 방법이고, PC는 채널 간에서 1x1 컨벌류션을 하는 방법이다.

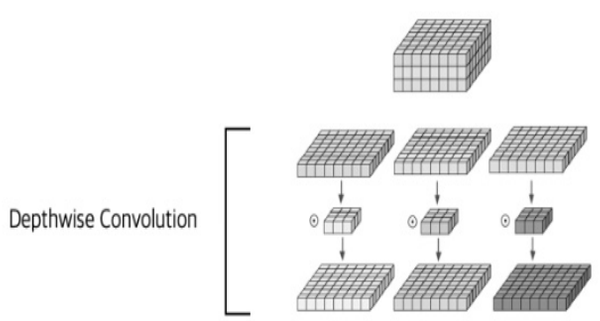

Fig. 8. Depthwise Convolution [7] Method

Mobilenet v2[3]는 Mobilenet v1의 속도 저하 원인이었던 PC의 부담을 인식하고 DC의 연산 비중을 늘리는 방법을 사용한다. Fig. 10에서 볼 수 있듯이 채널 간에서 확장층(Expansion Layer), 투영층(Projection Layer)이 추가되었고,

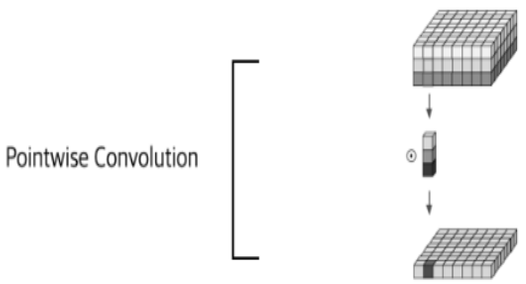

Fig. 9. Pointwise Convolution [7] Method

그 중간에 DC가 존재한다. 즉, 확장층 PC에서 채널(Channel)을 늘려준 상태에서 DC를 한다. 채널을 확장하게 되면 채널의 깊이가 깊어져 저차원의 종속공간(low-dimensioal subspace)으로 내장(embedding)되는 경우도 비선형 변환(ReLU)에 의한 정보 손실이 적어진다. 투영층에서는 원래의 채널 개수로 감소시켜주는 역할을 하게 된다.

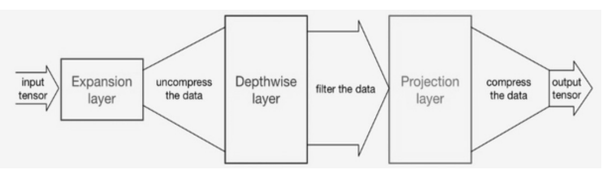

Fig. 10. Mobilenet V2 [3] Structure

3. 실험 및 고찰

본 연구에서의 접근방식인 SSD 알고리즘을 적용하여 성능을 비교 평가하기 위한 실험 환경은 다음과 같다. 본 논문에서는 Tensorflow Object Detection API를 이용하며, 실시간 처리가 가능한 플랫폼으로는 8GB 128bit LPDDR의 메모리를 장착한 NVIDIA Jetson TX2 임베디드 보드를 선택하여 실험 환경에 맞게 섬세한 조율(Fine-Tuning)을 하였다. TF-TRT와 TRT의 결과를 비교 및 검토하기 위하여 아래와 같은 실험들을 진행하였다.

우선 검출에 사용할 모델을 선정하였다. SSD를 이용한 검출 모델은 다양하며 대표적으로 Mobilenet과 Inception이 있다. 그 중 Mobilenet v2와 Inception v2를 이용하고, 데이터셋은 COCO 데이터셋(Dataset)을 이용하여 NVIDIA의 젯슨 추론(Jetson Inference) [14] 코드로 같은 조건에서 실시간 객체 검출 성능을 수행하였을 때, 성능 비교 결과는 Fig. 11에서 볼 수 있듯이 Mobilenet v2의 속도가 Inception v2보다 빠른 것을 확인할 수 있었다. 임베디드 환경에서 객체 검출을 수행할 때 속도는 중요한 결과 중 하나이기 때문에 검출 모델은 Mobilenet v2를 선정하였다.

이후, 개방형 데이터셋(Open Dataset)인 WIDER FACE Dataset을 사용하여 사전 학습된 SSD Mobilenet v2를 API를 이용하여 Fine-Tuning을 진행하였다. 약 12,000개의 학

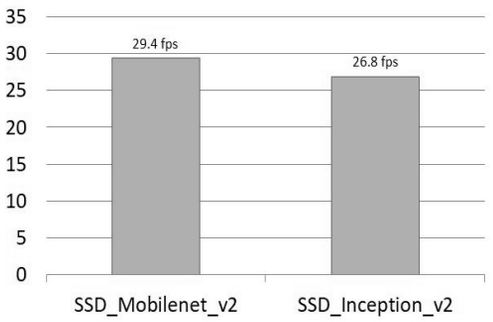

Fig. 11. Performance Comparison between Mobilenet and Inception(FPS)

습용 데이터셋(Training Dataset)과 200개의 테스트 데이터셋(Test Dataset)으로 훈련을 진행하였다. 데이터셋의 개수, 훈련 횟수 및 다른 조건들은 모두 동일하게 설정한 후 batch size에 따른 훈련 결과를 비교하였다. 해당 실험은 임베디드 보드인 Jetson TX2에서 진행하였다.

Table 1. Training Results According to Batch Size

	Accuarcy	FPS
Batch size = 32	1	1
Batch size = 24	0.92	1
Batch size = 16	0.84	1

batch size는 32, 24, 16의 3가지의 경우로 나눠 실험을 진행하였다. batch size가 32일 경우를 기준으로 두고 정확도와 속도 모두 사진 100장의 평균을 계산한 결과를 기재하였다. Table 1에서 볼 수 있듯이 속도는 batch size에 크게 영향을 받지 않았으나 정확도에서 차이를 볼 수 있었다. batch size가 작을수록 경계 상자가 한 객체에 여러 개가 그려지거나 객체의 정확도가 낮은 결과를 보였다.

Fig. 12. Face Detection Result

위와 같은 결과로 훈련 모델은 SSD Mobilenet V2를 사용하고 중요한 파라미터인 batch size의 크기는 32로 결정을 하였다. 해당 모델로 Fine-Tuning하고 얼굴 검출 시험을

시도한 결과 Fig. 12와 같이 6개의 다양한 얼굴 모양에 대해 모두 정확하게 검출하는 결과를 얻었다.

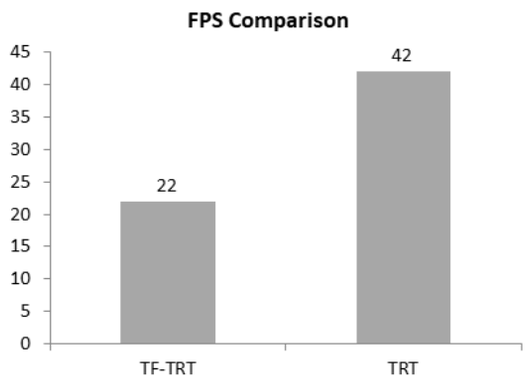

Fig. 13. Comparison between TF-TRT and TRT

Fig. 13은 Fine-Tuning 한 새로운 모델을 TF-TRT [11]와 TRT [12]로 각각 최적화한 후 처리속도(FPS)를 비교한 결과이다. 그래프 일부분만 최적화를 수행하는 TF-TRT보다 전체 그래프를 최적화하는 TRT가 빠른 것을 알 수 있다. 같은 영상에 대해 TF-TRT로 최적화를 한 결과는 평균적으로 처리속도가 22 FPS로 산출되었으며, TRT로 최적화를 한 결과는 평균적으로 42 FPS의 결과를 보였다.

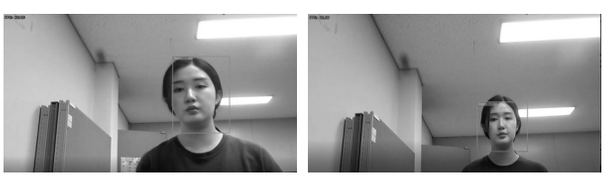

Fig. 14. Additional New Face Object Detection Results

Fig. 14는 새롭게 훈련한 얼굴 모델로 Jetson TX2에서 실시간 검출을 한 결과이다. 임베디드 보드에서도 빠르고 정확하게 검출이 되는 것을 확인할 수 있었다.

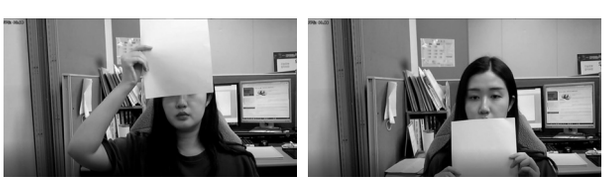

Fig. 15. Experimental Results on Partial Screening Images

Fig. 15의 왼쪽 사진처럼 눈이나 오른쪽 사진처럼 입을 일부 가리게 되면 얼굴이 정상적으로 검출이 되지 않는 것을 확인할 수 있다. 즉, 눈, 코와 입 모두 나와야 얼굴 영역을 인식하고 검출을 하게 되는 것을 확인하였다. 얼굴의 일부가 가려져도 성공적으로 검출이 되기 위해서는 더 다양한 경우의 데이터 세트를 확보한 경우의 학습과 테스트를 수행하여야 한다.

본 연구에서의 분석과 실험을 통해 여러 파라미터들을 조정해 본 결과, 속도에 대한 개선의 요인은 알고리즘 종류, 모델 종류, TensorRT 엔진의 사용 여부가 제일 큰 영향을 미쳤고, 정확도의 개선은 batch size의 변화가 가장 큰 관계가 있다는 것을 확인하였다.

4. 결론 및 향후 연구계획

본 논문을 통해 연산 가속 기능이 있는 임베디드 보드를 사용한 고속 플랫폼 환경에서 얼굴객체 검출을 더욱 효과적으로 구현하고 정확도와 속도 등의 성능을 기존의 방식과 비교하여 실시간 활용에 적합한 딥러닝 네트워크의 구조를 검토하고 제시하였다. 또한, 실시간 활용을 위한 새로운 접근방법의 성능 개선 가능성과 효과를 실험적으로 보여주는데 필요한 실험을 수행하였다. 기존 Tensorflow만을 이용한 객체 검출은 임베디드 보드에서 사용하기에는 속도가 느려 실시간성이 보장되지 않았으며 또한, YOLO보다 느린 속도의 SSD 알고리즘을 선택하여 TensorRT 채용의 효과를 비교해서 강조할 수 있도록 하였다. 추가로 같이 사용할 객체 검출 모델도 실험을 통해 다른 방식에 비해 약 3 FPS가 빠른 SSD Mobilenet V2를 선정할 수 있었다. 검출 모델을 결정한 이후 다양한 조건별 실험을 통해 batch size를 조절하여 비슷한 속도에서 더 높은 정확도를 가질 수 있게 하였다. 그 결과로 batch size의 변화는 속도의 영향을 거의 받지 않지만, 정확도에서 차이가 나는 것을 확인하여 batch size의 값을 32로 결정하였다. 하지만 임베디드 보드에서는 가장 빠른 속도를 보장하는 모델을 사용하여 검출해도 10 FPS 미만의 속도가 나왔다. 이러한 속도에 대한 보완방법으로 TensorRT를 이용하여 40 FPS 이상의 속도를 보장받음으로써, 임베디드 보드 환경에서 빠른 속도와 높은 정확도로 객체 검출을 할 수 있게 됨을 확인하였다.

향후 연구는 부족한 정확도를 올리기 위한 내용을 진행할 필요가 있으며 데이터 세트의 개수 및 다양성을 개선하여 얼굴의 일부 영상에 대해서도 검출률을 향상하며, 훈련 과정에서의 여러 파라미터들을 조정하여 모델 자체의 성능을 향상하는 연구도 요구된다. 또한, TensorRT는 다른 주요 프레임에 대해서도 사용할 수 있어 다른 알고리즘과 다양한 딥러닝 네트워크를 활용한 비교 실험도 가능하다.

References

[1] Zhengxia Zou, Zhenwei Shi, Member, IEEE, Yuhong Guo, and Jieping Ye, Senior Member, IEEE - Object Detection in 20 Years: A Survey.

[2] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg1/ UNC Chapel Hill Zoox Inc. Google Inc. University of Michigan, Ann-Arbor - SSD: Single Shot MultiBox Detector.

[3] Mark Sandler, Andrew Howard Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen/Google Inc. - MobileNetV2: Inverted Residuals and Linear Bottlenecks.
[4] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton - ImageNet Classification with Deep Convolutional Neural Networks.
[5] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun - Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.
[6] CV-Tricks.com-Learn Machine Learning, AI & Computer Vision - Object-detection/Faster-r-cnn-yolo-ssd.
[7] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam - MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.
[8] "Face Recognition on Jetson Nano using TensorRT," https://towardsdatascience.com/face-recognition-using-tensorrt-on-jetson-nano-set-up-in-less-than-5min-7c00bf730085
[9] "NVIDIA TensorRT Documentation," https://docs.nvidia.com/deeplearning/tensorrt/developer-guide/index.html
[10] Joseph Redmon(University of Washington), Santosh Divvala (Allen Institute for Artificial Intelligence), Ross Girshick (Facebook AI Research), Ali Farhadi(University of Washington) - You Only Look Once: Unified, Real-Time Object Detection.
[11] https://github.com/jkjung-avt/tf_trt_models
[12] https://github.com/jkjung-avt/tensorrt_demos
[13] https://github.com/AastaNV/TRT_object_detection
[14] https://github.com/dusty-nv/jetson-inference

유 혜 빈
https://orcid.org/0000-0003-1465-8447
e-mail : gpqls7662@hknu.ac.kr
2017년~현 재 한경대학교 ICT로봇기계공학부 학사과정
관심분야 : Image Processing, Embedded System Machine Learning

박 명 숙
https://orcid.org/0000-0003-1048-0792
e-mail : nicems@nate.com
2016년 한경대학교 전기전자제어공학과 (석사)
2019년 한경대학교 ICT로봇기계공학부 박사수료
관심분야 : Image Processing, Robot Control, Machine Learning

김 상 훈
https://orcid.org/0000-0003-1048-0792
e-mail : kimsh@hknu.ac.kr
1999년 고려대학교 전자공학과(박사)
2004년 ~ 2005년 University of Marvland, Colleage Park, Visiting Professor
1999년 ~ 현 재 한경대학교 ICT로봇기계공학부 교수
관심분야 : Image Processing, Robot Vision, Embedded System