https://doi.org/10.21289/KSIC.2024.27.4.1001


# 실시간 기반 매우 작은 객체 탐지를 위한 딥러닝 알고리즘 개발

# Development of a Deep Learning Algorithm for Small Object Detection in Real-Time

여우성[1], 박미영[2*]

Wooseong Yeo[1], Meeyoung Park[2*]

〈Abstract〉

Recent deep learning algorithms for object detection in real-time play a crucial role in various applications such as autonomous driving, traffic monitoring, health care, and water quality monitoring. The size of small objects, in particular, significantly impacts the accuracy of detection models. However, data containing small objects can lead to underfitting issues in models. Therefore, this study developed a deep learning model capable of quickly detecting small objects to provide more accurate predictions. The RE-SOD (Residual block based Small Object Detector) developed in this research enhances the detection performance for small objects by using RGB separation preprocessing and residual blocks. The model achieved an accuracy of 1.0 in image classification and an mAP50-95 score of 0.944 in object detection. The performance of this model was validated by comparing it with real-time detection models such as YOLOv5, YOLOv7, and YOLOv8.

*Keywords : Object Detection, Anomaly Detection, Image Classification, Deep Learning, Big Data*

1 정회원, 제1저자, 경남대학교 컴퓨터공학부, 학부연구생
E-mail: 202211831@student.kyungnam.ac.kr
2* 정회원, 교신저자, 경남대학교 컴퓨터공학부, 조교수
E-mail: mpark@kyungnam.ac.kr

1 First Author, Undergraduate Student, Dept. of Computer Engineering, Kyungnam University
2* Corresponding Author, Dept. of Computer Engineering, Kyungnam University

## 1. 서 론

객체 탐지는 이미지 분류의 확장 개념으로 딥러닝 알고리즘을 통해 이미지나 영상 속 객체를 감지하고 위치를 파악함으로써, 자율 주행 자동차, 교통 모니터링, 토양 상태 분석, 광산 및 채굴 비축양 측정, 수질, 식물 질병 탐지, 안면인식 등 다양한 방면에 활용되고 있다[1-3]. 이미지상 객체 크기는 탐지 모델 정확도에 직접적인 영향을 미치는데 작은 객체는 모델에 충분한 객체 정보 전달이 불가능하다. 이런 데이터로 학습한 객체 탐지 모델은 낮은 성능 즉, 언더피팅이 발생한다.

작은 객체는 경우에 따라 정의가 달라지는데 COCO dataset[4]의 경우, 32x32 픽셀 미만 일 때, Krishna et al.[5]의 경우 전체 이미지에서 객체가 차지하는 비율이 1% 미만일 때 작은 객체라 정의한다. 연구에 사용한 이미지의 객체는 평균적으로 22x22 픽셀로 약 0.05%의 이미지 공간을 차지하므로 작은 객체로 볼 수 있다.

객체 탐지는 영역 제안(Regional Proposal) 사용 방식에 따라 one-stage와 two-stage 탐지기 방식으로 나뉜다[6]. Two-stage는 영역 제안으로 객체 포함 가능성이 있는 후보 영역 집단을 추출한 뒤 객체를 분류한다. 영역 제안과 분류, 총 2단계로 나누어 탐지를 수행하기 때문에 one-stage에 비해 속도는 느리지만, 더 정확한 예측 결과를 제공한다. 대표적인 two-stage 모델로는 RCNN[6], Fast RCNN[7], Faster RCNN[8] 등이 있다. One-stage는 격자로 이미지를 셀로 나누어 영역 제안을 수행하고 각 셀에서 객체 존재 여부와 클래스를 예측한다. 영역 제안과 분류를 동시에 수행하기 때문에 신속한 탐지가 가능하지만, two-stage에 비해 정확도는 떨어지는 편이다. 대표적인 One-stage 모델은 YOLO(You Only Look Once)[9], SSD (Single Shot MultiBox Detector)[10], RetinaNet[11]이 있다. 본 연구에서 개발한 모델은 영역 제안을 사용하지 않기 때문에 두 카테고리 중 어디에 속하는지 정의하기 어렵다. 그러나 복잡한 과정 없이 한 번에 빠른 예측이 가능한 점에서 one-stage 탐지기에 근접하다고 볼 수 있다. 객관적인 평가를 위해 실시간 탐지에 특화된 YOLOv5, YOLOv7[12], YOLOv8을 만든 모델과 비교하였다.

본 연구에서는 ResNet에서 처음 도입된 Residual 개념을 모델 구조에 적용시켜 RE-SOD ( Residual block based Small Object Detector)를 개발하였다. 또한, 작은 객체에 대한 탐지 성능 향상을 위해 객체 비율을 늘리는 RGB를 분리하는 전처리 기법을 사용하였다. RE-SOD 는 이미지 분류에서 정확도 1.0, 객체 탐지에서 mAP50-95 0.944의 성능을 나타내었다. 최종적으로 실시간 객체 탐지에 특화된 YOLOv5, YOLOv7, YOLOv8과 RE-SOD 성능을 비교하여 개발한 모델 성능이 타당함을 입증하였다.

## 2. 데이터 수집 및 분석

### 2.1 데이터 준비 및 시각화

본 연구를 위해 Jang et. al[14]의 199장의 PNG 형식 깔따구 유충 데이터를 사용하였다. 모든 원본 이미지는 1128x844 픽셀로 22x22 픽셀 이하의 작은 객체를 포함한다.

작은 객체 문제를 해결하기 위해서 객체 너비, 높이가 32 픽셀을 넘도록 이미지를 확대하거나 객체가 전체 이미지의 1% 이상을 차지하도록 객체를 포함한 상태에서 배경 일부를 잘라내야 한다. 확대로 인한 크기 조정은 이미지 내부 특징과 패턴을 변형시킬 수 있고, 공간, 시간 복잡도를 증가시킨다[15]. 자르기 또한 경계 영역의 정보

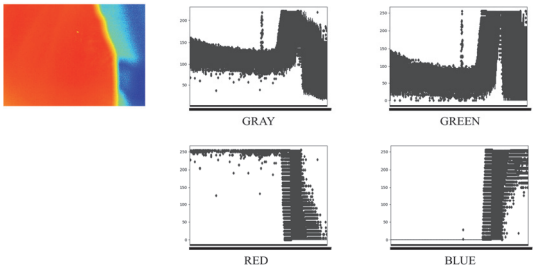



Fig. 1 Boxplots for grayscale(GRAY) and RED, GREEN, BLUE color channels

변형 가능성이 있지만, 연구에 사용한 데이터는 배경 특징이 단순하다. 그러나, 노이즈가 다수 간섭했기 때문에 객체를 중심으로 배경을 잘라내는 것이 노이즈 제거와 이미지 처리 속도를 높이는데 효과적이다. 따라서 이미지 내 객체 비율 1%를 목표로 객체를 중심으로 특정 크기로 이미지를 자르기 위한 분석을 수행하였다.

데이터 분포 파악을 위해, 이미지를 gray-scale(G)로 불러들여 boxplot으로 시각화하였다. 추가적으로, 원본 이미지가 Red(R), Green (G), Blue(B) 3가지 채널로 색상을 나타내는 컬러 이미지인 것을 고려해 RGB 각각을 분리해 시각화하였다. Fig. 1에서 나타난 바와 같이, Green은 G를 위아래로 늘린 형태로 데이터 분포 폭이 넓어 배경과 객체를 명확히 구분하였다. R과 B는 x-축을 대칭시킨 형태로 유사한 분포를 보였고, 실제 이미지상의 작은 객체는 나타나지 않았다.

### 2.2 데이터 전처리

먼저, 객체를 추출하기 위하여 RGB 분리를 통해 파악한 객체 위치를 원본에 대입하여 이미지를 잘라내었다. 22x22 객체는 이미지 크기가 220x220일 때, 정확히 이미지의 1%를 차지한다. 따라서, 그보다 작거나 근접하게 이미지를 잘라낼 필요가 있는데, ILSVRC(ImageNet Large Scale Visual Recognition Challenge)에서 우수한 성적을 거둔 AlexNet[16], VGGNet [17], GoogLeNet[18], ResNet[13]의 입력 방법을 바탕으로 224x224 크기의 이미지를 199장 중 172장을 잘라내었다.

이미지 172장 만으로는 뛰어난 탐지 성능을 기대하기 어렵다. 따라서, 전처리 데이터에 4배 증강(회전, 상하좌우 이동, 수평 반전)을 적용한 688개의 데이터를 생성하였다. 추가로, 객체가 없는 좌측 모서리에서 정상 데이터 172개를 습득했다. 배경만 존재하는 정상 데이터에 증강을 적용하면, 오히려 이미지 품질이 저하되었기 때문에 증강 대신 4배 복제를 통해 객체 이미지와 같은 688장을 만들어 데이터 불균형을 사전에 방지하도록 했다. 최종 입력 데이터는 객체와 정상 데이터를 더한 1,376장이다.

## 3. RE-SOD 수행

### 3.1 이미지 레이블링

객체 탐지 모델은 학습을 위해 이미지의 클래스 정보와 BBox(Bounding Box) 위치 정보를 필요로 한다. 본 연구에서는 LabelImg를 사용해 YOLO 형식(클래스 정보, x 중심, y 중심, 너비, 높이)의 레이블을 생성했다.

원활한 이미지, 레이블 정보 관리를 위해 이미지 경로와 클래스, x 중심, y 중심, 너비, 높이 정보를 매핑한 csv 파일을 생성했다. 객체 이미지는 클래스 정보 1과 0~1사이의 BBox 값을 가지고 정상 이미지는 클래스와 BBox를 값에 0을 넣어 모델이 배경만 있는 상황에는 BBox를 생성하지 않도록 했다.

### 3.2 Residual Block

일반적으로 모델의 깊이를 깊게 가질수록 기울

기가 사라지거나 폭주하는 문제가 발생한다. 2015 ILSVRC의 ResNet은 잔차(Residual) 개념을 처음으로 도입한 모델로 기울기 소실 문제를 해결하고 깊은 네트워크 구조를 구축함으로써 뛰어난 성능을 달성할 수 있었다.

Fig. 2(a)와 같이, ResNet의 기본 잔차 블록은 한 개 이상의 가중치 레이어를 건너뛰어 연결하는 스킵 연결(Skip-connection)을 중심으로 학습된다. 스킵 연결은 하단 레이어에 입력을 직접 전달하므로 깊은 레이어 통과 시 발생할 수 있는 정보의 손실을 줄인다. 확장된 잔차 블록인 경우, 합성곱(Convolution) 연산을 건너뛴 입력 x를 출력

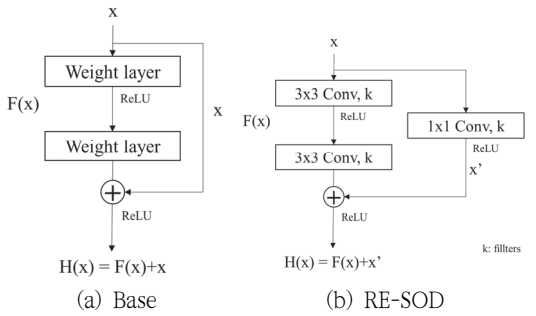



**Fig. 2 Architecture of Residual Block**

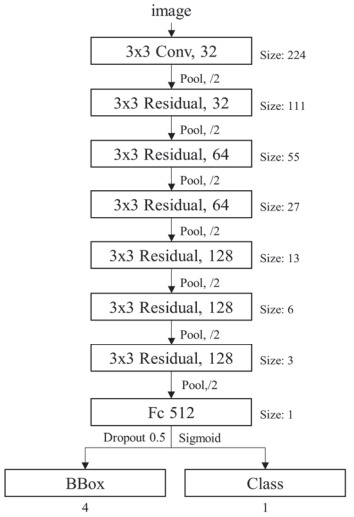



**Fig. 3 RE-SOD overall model structure**

F(x)에 연결한다. 이때, 최종 출력 H(x)는 F(x)+x'가 되고 이 상태에서 네트워크는 입력과 출력 사이의 차이인 잔차를 학습하게 된다[13].

RE-SOD에 사용한 잔차 블록은 Fig. 2(b)에서와 같이, 기본 가중치 레이어에 3x3 합성곱 레이어를 배치한 형태이다. 스킵 연결을 통해 출력에 입력을 더하는 개념은 기존의 것과 동일하다. 입력 x에 추가적인 1x1 합성곱 레이어로 x'을 연산하였다. ResNet의 잔차 블록에서도 이러한 방식으로 x 차원을 축소하여 연산 효율을 높였다. 본 논문은 x'에 비선형 활성화 함수(ReLU)를 적용하여 네트워크의 표현력을 한층 더 높였다. 실험 결과, x'에 ReLU를 적용한 모델이 그렇지 않은 모델보다 더 높은 성능을 보였다. 최종적으로 x'에 F(x)를 더한 F(x)+x'이 잔차 블록의 출력이다.

Fig. 3에서 나타낸 것과 같이, RE-SOD 모델의 전체 구조는, 224x224x3 컬러 이미지를 입력받고, 잔차 블록을 중심으로 객체 특징을 추출한다. 합성곱 또는 잔차 연산마다 풀링(Pooling)을 배치하여 점진적으로 연산 복잡도를 감소시켰다. 잔차 블록, 풀링을 추가함에 따라 모델 성능이 증가하였기 때문에 풀링으로 인해 이미지 사이즈가 1x1이 될 때까지 잔차 블록을 배치하였다. 따라서 합성곱 레이어 1개, 잔차 블록 6개, 풀링 7개를 사용하였다.

활성화 함수는 ReLU, 커널 크기는 3x3을 사용하였다. 필터 수는 32와 64를 2번, 128개를 3번 사용하여 단순한 패턴에서 추상적인 패턴을 고르게 추출하였다. 합성곱 연산이 모두 끝나면 FC(Fully-Connected) 레이어에서 3차원 정보를 1차원으로 변환하고 512개의 뉴런으로 추출된 패턴을 학습하였다. 이때 드롭아웃(Dropout) 비율을 0.5로 설정하여 과적합을 방지하였다. 마지막으로 Sigmoid 함수를 통해 BBox 회귀, Class 분류 작업을 동시에 수행한다.

Class 분류 작업에서는 이진 분류에 특화된 이

진 교차 엔트로피(Binary cross-entropy)를 사용하였다. 일반적으로 사용되는 손실함수는 MSE (Mean Squared Error), MAE(MeanAvera ge Error), Huber[19] 등이 사용된다. Huber 손실함수는 MSE와 MAE를 절충한 함수로서 특정 임계값 delta($\delta$)를 조정하여, 오차 a가 $\delta$보다 작으면 MSE, 크면 MAE와 유사한 함수를 적용한다. 본 연구에서는 데이터 특성에 최적화된 손실함수를 적용하기 위해 Huber 손실함수를 사용하였다.

## 4. RE-SOD 수행 결과

RE-SOD를 학습 후 베스트 모델을 선정하기 위해 $\delta$(0.2, 0.3, 0.4), epoch(200, 300, 500), learning rate(L) (0.01, 0.001, 0.0001, 0.00001)에 대한 하이퍼 파라미터 튜닝을 진행하였다. 실험을 통해 Batch size는 32로 고정하였고, optimizer는 안정적이고 빠른 학습에 효과적인 Adam을 사용하였다. 사전학습된 YOLO 중 최적의 파라미터를 찾기 위해, v5와 v8은 파라미터 Nano(n), Small(s), Medium(m), Large(l), Xla rge(x)를, v7은 Tiny(v7-tiny), Normal(v7), Xlarge(v7x)을 사용하여 epoch(200, 300, 500) 별로 학습을 진행하였다.

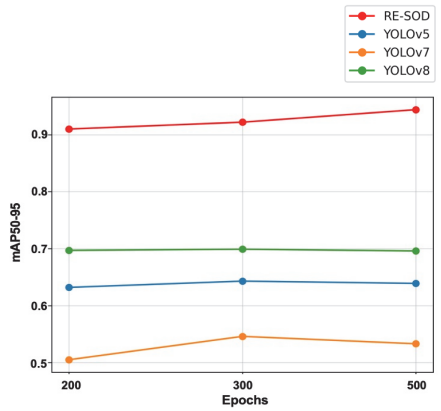
성능지표로는 객체 탐지에 대표적인 mAP(mean Average Precision) 50과 mAP50-95로 최적의 모델을 선정하였다. mAP는 실제 BBox와 예측 BBox 간의 교차 영역인 IoU(Intersection over Union) 특정 임계값을 기준으로 계산된다. mAP50은 클래스별로 IoU가 50% 이상인 경우에만 계산한 AP(Average Precision)들의 평균이다. mAP50-95는 IoU 50%에서 95%까지 5%씩 늘려가며 계산한 mAP들의 평균으로 mAP50보다 엄격한 평가 지표이다.

학습 수렴 속도 분석을 위해 epoch 별로 최적의 모델을 선정하였다. Table 1은 RE-SOD와 YOLO 모델들을 epoch 별 mA50-95 성능과 예측에 걸린 시간을 함께 비교한 것이다. RE-SOD의 경우, epoch 200일 때, $\delta$= 0.2, L=0.001인 모델이 정확도: 1.0, mAP50-95: 0.910을 달성하였다. Epoch 300일 때, $\delta$=0.4, L=0.0001인 모델이 정확도: 1.0, mAP50-95: 0.922를 달성하였다. Epoch 500일 때, $\delta$=0.2, L=0.0001인 모델이 정확도: 1.0, mAP50-95: 0.944를 달성하였다. 추가적으로, RE-SOD와 비교한 YOLO 모델들의 epoch 별 성능은 다음과 같다. Epoch 200일 때, YOLOv5: mAP50-9 5 0.632, YOLOv7: mAP50-95 0.505, YOLOv8: mAP50-95 0.697을 달성하였다. Epoch 300 일 때, YOLOv5: mAP50-95 0.643, YOLOv7: mAP50-95 0.546, YOLOv8: mAP50 -95 0.699을 달성하였다. Epoch 500 일 때, YO LOv5: mAP50-95 0.639, YOLOv7: mAP50-95 0.533, YOLOv8: mAP50-95 0.696을 달성하였다.

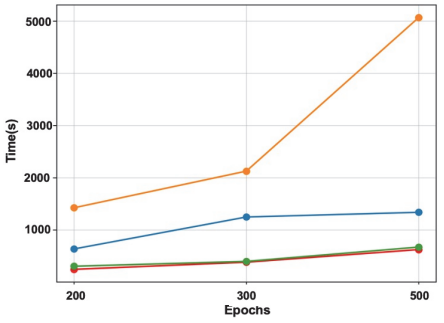
Fig. 4는 모델들의 성능 비교를 위해 epoch 별

**Table 1. Performance comparison between YOLO models and the RE-SOD**

| Epoch | Model | mAP50-95 | time(s) |
|---|---|---|---|
| 200 | YOLOv5 | 0.632 | 637 |
| | YOLOv7 | 0.505 | 1,425 |
| | YOLOv8 | 0.697 | 306 |
| | **RE-SOD** | **0.910** | **247** |
| 300 | YOLOv5 | 0.643 | 1,249 |
| | YOLOv7 | 0.546 | 2,127 |
| | YOLOv8 | 0.699 | 399 |
| | **RE-SOD** | **0.922** | **384** |
| 500 | Yolov5 | 0.639 | 1,339 |
| | Yolov7 | 0.533 | 5,068 |
| | Yolov8 | 0.696 | 673 |
| | **RE-SOD** | **0.944** | **626** |




(a) mAP50-95 per epoch




(b) Training time per epoch

**Fig. 4 mAP50-95 and Training time per epoch**

최적 모델의 mA50-95 (Fig. 4(a))와 학습에 걸린 시간(Fig. 4(b)에 대한 결과를 보여준다. YOLO 모델 중에서는 YOLOv8이 가장 높은 mAP50-95와 학습 시간을 보인다. 또한 YOLOv7이 가장 긴 학습시간과 낮은 성능을 보였다. YOLOv5는 v7보다 높으나 v8보다는 낮은 성능을 기록했다. RE-SOD는 가장 높은 mAP50-95와 YOLOv8보다 빠른 학습 시간을 나타내었다.

## 5. 결 론

최근 실시간 객체 검출을 위한 딥러닝 알고리즘은 자율 주행, 교통, 의료, 수질 및 환경 모니터링 등 다양한 산업 분야에서 활용되고 있다. 특히 수질 모니터링에서 유충과 같은 작은 객체의 탐지는 실시간 검출 모델의 정확도가 매우 큰 영향을 미친다. 따라서 본 연구에서는 실시간에 매우 작은 객체를 더욱 정확하고 빠르게 예측하기 위해 ResNet 기반 RE-SOD 딥러닝 모델을 개발하였다. RE-SOD는 1.0의 정확도와 기본 YOLO 모델을 그대로 사용하는 것에 비하여 매우 빠른 속도로 객체를 검출함을 보였다. 향후 연구로는 RE-SOD가 더 복잡한 배경을 포함한 다른 데이터셋에서도 작은 물체를 탐지하는데 높은 탐지 성능을 보일 수 있는지에 대한 추가 실험을 진행하고자 한다.

## 사 사

이 성과는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. RS-2023-00252141), 또한 2023년도 교육부의 재원으로 한국연구재단의 지원을 받아 수행된 지자체-대학 협력기반 지역혁신 사업(2021RIS-003)의 결과임.

## 참고문헌

[1] J. Song, S. Lee, and A. Park, “A study on the industrial application of image recognition technology,” The Journal of the Korea Contents Association, vol. 20, no. 7, pp. 86-96, (2020).

[2] N. J. Tahira, J. Park, S. Lim, J. Park, “YOLOv5 based Anomaly Detection for Subway Safety Management Using Dilated Convolution,” Journal of the Korean Society of Industry Convergence, vol. 26, no. 2_1, pp. 217-223, (2023).

[3] T. Jin, “Color Pattern Recognition and Tracking for Multi-Object Tracking in Artificial Intelligence Space,” Journal of the Korean Society of Industry Convergence, vol. 27, no. 2_2, pp. 319-324, (2024).
[4] T. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. Zitnick, “Microsoft coco: Common objects in context,” Proc. of the Computer Vision–ECCV 2014, pp. 740-755, (2014).
[5] H. Krishna, and C.V. Jawahar “Improving small object detection,” Proc. of the 4th IAPR Asian conference on pattern recognition (ACPR), pp. 340-345, (2017).
[6] R. Girshick, J. Donahue, T. Darrell, J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” Proc. of the IEEE conference on computer vision and pattern recognition, pp. 580-587, (2014).
[7] R. Girshick, “Fast r-cnn,” Proc. of the IEEE international conference on computer vision, pp. 1440-1448. (2015).
[8] S. Ren, K. H. Ross, G. J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” Proc. of the Advances in neural information processing systems, vol. 28, (2015).
[9] J. Redmon, S. Divvala, R. Girshick, A.Farhadi, “You only look once: Unified, real-time object detection,” Proc. of the IEEE conference on computer vision and pattern recognition, pp. 779-788, (2016).
[10] L. Wei, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” Proc. of the Computer Vision–ECCV 2016: 14th European Conference, pp. 21-37, (2016).
[11] T. Lin, P. Goyal, R. Girshick, K. He, P. Dollar, “Focal loss for dense object detection,” Proc. of the IEEE international conference on computer vision. pp. 2980-2988, (2017).
[12] W. Chien-Yao, A. Bochkovskiy, and H. M. Liao, “YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors,” Proc. of the IEEE/CVF conferenc e on computer vision and pattern recognition, pp. 7464-7475, (2023).
[13] H. Kaiming, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” Proc. of the IEEE conference on computer vision and pattern recognition, pp. 770-778, (2016).
[14] G. Jang, W. Yeo, M. Park, and Y. Park, “RT-CLAD: Artificial Intelligence-Based Real-Time Chironomid Larva Detection in Drinking Water Treatment Plants,” Sensors vol. 24, no. 1, pp 177, (2023).
[15] H. Mahdi, “Enlarging smaller images before inputting into convolutional neural network: zero-padding vs. interpolation,” Journal of Big Data, vol. 6, no. 1, pp. 1-13, (2019).
[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “Imagenet classification with deep convolutional neural networks,” Proc. of the Advances in neural information processing systems, pp. 1-9. (2012).
[17] K. Simonyan, and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” Proc. of the ICLR 2015, arXiv: 1409.1556 (2014).
[18] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, Dumitru Erhan, V. Vanhoucke, A. Rabinovich, “Going deeper with convolutions,” Proc. of the IEEE conference on computer vision and pattern recognition, pp. 1-9. (2015)
[19] P. J. Huber, “Robust estimation of a location parameter,” Breakthroughs in statistics: Methodology and distribution. New York, NY: Springer New York, pp. 492-518, (1992).

(접수: 2024.07.31. 수정: 2024.08.05. 게재확정: 2024.08.08.)