[UCI]I804:48009-200000507467

공학석사 학위논문

증식 데이터셋 구축을 통한 객체 인식 성능 향상 방안 연구

Improvement of Object detection by developing framework of data augmentation

울 산 대 학 교 대 학 원
전기전자정보시스템공학과
이 동 희

증식 데이터셋 구축을 통한 객체 인식 성능 향상 방안 연구

지 도 교 수 김 병 우

이 논문을 공학석사학위 논문으로 제출함

2021년 08월

울 산 대 학 교 대 학 원
전기전자정보시스템공학과
이 동 희

이 동 희의 공학석사학위 논문을 인준함

심사위원장 김 한 실 인

심사위원 김 병 우 인

심사위원 곽 수 진 인

울 산 대 학 교 대 학 원

2021년 08월

감사의 글

연구실에 들어와 대학원에 입학 한지가 엊그제 같은데 어느덧 졸업논문을 쓰고 있습니다. 부족했던 제가 성숙하고 더 발전한 모습으로 성장 할 수 있었던 이유는 제 주변의 많은 분들의 관심과 격려가 있어서라고 생각합니다. 이 자리를 빌려 저에게 많은 격려와 응원을 해주신 모든 분들께 감사의 마음을 전하며, 이 마음 잊지 않고 간직하겠습니다.

우선 지난 2년 동안 부족했던 저를 지도해주신 김병우 교수님께 감사의 마음을 전해드리고 싶습니다. 많은 경험을 쌓을 수 있도록 기회를 주신 덕분에 새롭고 다양한 경험과 부족함 없는 성과를 달성할 수 있었습니다. 생활하면서 교수님께서 지혜와 가르침은 앞으로의 삶을 올바르게 해쳐나가기 위해 꼭 필요한 밑거름이 될 것입니다. 또한, 바쁘신 와중에도 시간을 내주시어 저의 부족한 석사학위논문을 심사해 주신 김한실 교수님과 곽수진 박사님께 감사의 말씀을 드립니다.

석사과정동안 함께 동고동락했던 자동차 전자제어 연구실의 선배들과 후배들에게도 감사의 마음을 전합니다. 석사과정이라는 시간 동안 많은 것을 배울 수 있었던 것은 연구실의 동료들이 있었기 때문이라고 생각합니다. 가족보다 더 많은 시간을 함께 보내며 서로 밀어주고 당겨주었던 시간을 잊지 못할 것이며, 앞으로도 소중히 간직하며 지내겠습니다. 랩장으로 연구실을 이끌며 누구보다 연구실을 생각하고 많은 조언을 해주고 다양한 분야의 지식을 공유해준 재우형, 엄마 같은 마음으로 따뜻한 조언과 공감을 통해 보이지 않는 곳에서도 많은 도움을 주는 경은이누나, 먼저 졸업하여 취직하여 직장을 다니고 있지만 저의 고충과 이야기를 자주 들어준 명이형과 득경이형, 항상 긍정적이고 연구실의 분위기를 밝게 만들어주는 정은이, 같은 기간 동안 동기로써 함께 많은 이야기를 나누며 생활한 종규, 한국말은 어눌하지만 한결같은 자세로 공부하며 배울게 많은 팽도형, 적극적인 자세로 열심히 공부하는 민식이를 포함하여 모두에게 감사의 인사를 드립니다. 또한, 바쁘신 데도 언제든지 조언과 응원을 해주신 자동차전자제어 연구실 선배님들께도 감사의 인사를 드립니다.

제가 지금까지 잘 성장할 수 있었던 것은 한결같은 신뢰로 끊임없이 지켜봐주던 가족이 있었기 때문입니다. 무사히 공부를 마칠 수 있도록 무한한 응원과 믿음을 보여주신 어머니, 아버지께 감사의 마음을 전합니다. 그리고 묵묵히 응원하면서 지켜봐주는 누나에게 고맙다는 말을 전합니다. 또한 힘들고 지칠 때마다 언제나 힘이 되어주고 멀리서 응원을 해준 친구들에게 감사의 말을 전합니다.

마지막으로 엇나가지 않고 지금의 저가 있기까지에는 저를 지켜봐주시고 응원해주신 모든 분들 덕분입니다. 다시 한 번 감사드립니다.

2021. 08. 13.

이동희 올림

증식 데이터셋 구축을 통한 객체 인식 성능 향상 방안 연구

이 동 희

울산대학교대학원 전기전자정보시스템공학부

요 약

최근 인공지능의 발달로 다양한 산업 분야에서 급격한 환경 변화가 일어나고 있다. 이미 많은 공장 라인에서는 사람을 대신하여 자동화장비 및 무인 로봇으로 대체되고 있으며, 그에 대한 수요는 증대하고 있다. 그 뿐만 아니라 자율주행 차량, 무인 택배 로봇 등과 같이 제한된 환경을 넘어 우리의 일상 반경의 경계도 허물어 많은 부분에서 차지하는 비중이 증가하고 있다. 따라서 이와 같이 다양한 변수를 지닌 환경에서 대응이 가능한 기술적 요구가 꾸준히 제기되고 예상치 못한 환경에서의 로봇 움직임은 강건해야 한다. 이에 핵심적인 기술로, 다양한 임무 수행이 가능한 로봇 플랫폼 구축과 딥러닝 기반의 영상 객체 인식 기술이 필수적이다.

현재 물체 인식 관련 분야에서 딥러닝 구조 개선, 딥러닝 학습 방식 최적화 등과 같은 연구가 꾸준히 진행중이다. 그 이외에도 데이터 증식을 통해 인식 성능 향상을 도모하는 연구도 상당히 진전되고 있다. 특히 물체 인지 기술의 안정적이고 좋은 성능을 위해선 편향되지 않으며 수십에서 수백만장의 방대한 데이터셋이 요구된다. 주로 온라인에서 제공되는 공유 데이터셋은 차량, 사람, 컵, 동물 등과 같이 보편적인 사물로 구성되어 있다. 그러나 산업이나 개인적으로 특수한 환경에서 볼 수 있는 사물에 대한 데이터셋은 얻기 어려울 뿐만 아니라 구축하기에는 상당한 시간과 인력이 소모된다.

본 논문에서는 특화된 환경에서 물체 인식의 성능 향상을 위한 데이터셋 구축 프레임 워크를 제시한다. 이를 검증하기 위해 딥러닝 학습을 통해 성능 평가를 실시하고 실제 환경에서 테스트를 통해 정확도를 검증한다. 사물 인식(Object detection)을 위한 딥러닝 학습에 필요한 커스텀한 데이터셋을 구축하였다. 먼저 공유 데이터셋에서 얻지

못하는 커스텀 데이터셋을 대상으로 온라인에서 데이터 수집을 실시하였고, 부족한 양과 편향된 데이터를 보완하기 위해 데이터 조작(Manipulation)과 딥러닝 GAN(Generative Adversarial Network)로 이루어진 프로세스를 구축하여 데이터 증식을 실시하였다. 이미지 Manipulation으로는 회전, 반전, 블러, 랜덤 노이즈를 적용하였고, 딥러닝 기반의 GAN은 DCGAN, WGAN, SRGAN 구조를 통해 학습을 실시하였다. 해당 기술들을 효율성과 성능 최적화를 위해 데이터 증식 프레임 워크를 구축하였다.

해당 데이터셋의 성능 검증을 위해 1-stage 객체인식 알고리즘인 YOLO 계열 구조로 학습을 실시하였다. 이는 정확도와 실시간 인식시간의 절충된 알고리즘으로 빠른 시간 내에 적절한 정확도 결과를 가져다 주기에 이를 적용하였다. 총 YOLOv4, YOLOv3, YOLOv3-tiny 구조 기반으로 딥러닝 학습을 실시하였다. YOLOv4 학습 시에 최신 딥러닝 학습 기법인 Mish activation function, 상호 연관성을 띄는 구역 Dropout을 실시하는 DropBlock, CmBN 등을 활용하여 학습을 진행하여 최적 모델을 생성하였다. 그 이외에도 데이터셋별로 YOLOv3, YOLOv3-tiny 구조로 학습 모델을 생성하였다.

성능 검증을 위해 Precision과 Recall의 인자를 포함하는 PR곡선의 면적을 계산하는 AP 평가지표를 활용하였다. 수집된 데이터셋, 증식 데이터셋별로 동일한 테스트 데이터셋으로 AP 정확도를 추출하여 성능을 검증하였고 다양한 YOLO 구조 기반으로 검증을 실시하였다. 제시한 증식 데이터셋은 기존 수집된 데이터셋 대비하여 mAP 기준으로 평균 10~20% 의 정확도 향상, 최대 36%의 성능 향상을 확인하였다. 각각의 증식 기법에 의해 증식된 이미지 개수 당 정확도 향상 비율을 분석하였다. 기존 Manipulation 기법에 비해 GAN 기반의 데이터는 새로운 이미지를 생성하면서 편향적이지 않는 데이터셋을 생성함을 알 수 있었다.

추가적으로 생성된 최적 모델을 이용하여 실제환경에서 임베디드 기반 물체 인식 성능 테스트를 진행하였다. 이 때 ROS 기반의 UGV의 플랫폼을 활용하여 물체 인식 성능을 확인하였다. 또한 UGV의 탑재된 임베디드 보드인 NANO, TX2, Xavier에 인지 정확도 및 속도 성능을 확인하였다. 인지하고자 하는 사물 주변에서 실시간 정확도를 학습 모델별로 확인하여 정확도 향상을 확인하였다. 또한 YOLOv3-tiny 기준으로 Xavier기반 평균 54FPS, TX2 기반 평균 24FPS, NANO 기반 평균 12FPS의 속도를 제공하였다. YOLOv3기준으로 Xavier기반 평균 14FPS, TX2 기반 평균 8FPS, NANO 기반 평균 3FPS의 속도를 제공하였다.

본 논문에는 커스텀한 사물의 데이터셋을 구축하기 위한 프레임 워크를 제시하였고, 이를 YOLO 계열의 구조로 딥러닝 학습을 실시하여 최적 모델을 생성하였다. 생성된

모델의 성능 검증을 AP기반 실시하였으며, 실제환경에서 데이터셋 종류에 따라 테스트를 실시하여 제시하는 데이터셋의 안정적이고 높은 성능을 실시간으로 확인하였다. 이 뿐만 아니라 물체 인식의 인지 속도를 개선하고자 다양한 임베디드 시스템을 이용하여 성능 평가를 실시하였다.
주요어 : 딥러닝(Deep learning), 사물 인식(Object detection), 이미지 증식(Image Augmentation), 생성적 대립 신경망(GAN, Generative Adversarial Network), ROS(Robot Operating System)

목 차

그림목차

표 목차

I. 서론

1. 연구 배경 및 필요성

현재 인공지능 기술 발달로 다양한 산업 분야에서 급격한 환경 변화가 일어나고 있다. 특히 자율 주행 관련 시장 규모는 2019년 기준 542억달러에서 2026년에는 5560억달러규모로, 연평균 39.47%의 성장 가능성을 보일 것으로 예상되고 있다. 이에 따라 대부분의 OEM 자동차 대기업과 테슬라 기업은 자율 주행 차량의 연구 개발에 상당한 투자를 진행 중이고, 자율 주행 연구를 진행 중인 업체들이 개발에 박차를 가하며 2020년~2030년까지 자율주행 자동차의 상용화를 목표로 급속히 기술 발전이 나아가고 있고, 세계 각국 또한 자율 주행 시대를 준비하여 관련 법률 및 도로 환경 개선 등을 준비하고 있다.

인공지능 기반의 영상 인식은 자율 주행 기술 외에도 로봇과 카메라 기반의 인식이 요구되고 있으며, 다양한 산업 분야에 인지 기술이 접목되고 있다. 이미 많은 공장 라인에서는 사람을 대신하여 자동화장비 및 무인 로봇으로 대체되고 있으며, 그에 대한 수요는 증대하고 있다. 아마존과 같은 대기업과 스타트업인 키위와 같은 기업들이 자율 배송 로봇의 운영을 진행 중이다. 여기서 영상 인식 기술은 이동 로봇, 스마트 팩토리의 로봇 암에 카메라를 탑재하여 목표물의 상태를 파악하여 분류하고 이를 제어하기까지의 타겟의 정보를 획득하는 핵심적인 역할을 하게 된다. 이와 같이 예상 가능한 환경 또는 반복적인 패턴 환경에서의 영상 인식 기술은 상당한 영향력을 보여준다.

그러나 최근 테슬라 기업의 주행 보조 시스템에 의해 주행되던 차량의 지속적으로 사고를 발생되는 것처럼 예상치 못한 환경에서의 수많은 사고 발생이 도사린다. 테슬라의 사고와 같이 자차 기준의 차량의 모습으로 분별하기 어려운 시야에서 차량으로 인지하지 못하고 그대로 충돌하는 일이 빈번히 발생이 반복되고 있다. 특히 이러한 인지 기술의 오류로 차량의 감속 모드를 시행조차 하지 않고 사고가 발생되기에 사고 유형은 탑승자의 사망까지 초래하는 상당히 위험도가 높아지는 경향을 띄고 있다. 이와 같이 다양한 변수를 지닌 환경에서 대응이 가능한 기술적 요구가 꾸준히 제기되고 예상치 못한 환경에서의 이동체 움직임은 안전성을 보장해야 한다. 이는 반복적인 업무의 수준을 넘어서 동적인 물체로 이루어지고 변화되는 패턴에 적응이 가능해야 되므로 안정적인 딥러닝 기술의 수요가 대두된다.

자율주행 차량, 무인 택배 로봇 등과 같이 제한된 환경을 넘어 우리의 일상 반경의 경계도 허물어 많은 부분에서 차지하는 비중이 증가함에 따라 딥러닝 기반의 영상 인식 기술은 기존의 이미지의 픽셀 단위 접근하는 클래식한 컴퓨터 비전 기술에 비해 상당히 많은 연구가 진행되고 있다. 특히 2015년 이미지 인식 경진대회인 ILSVRC(ImageNet Large Scale Visual Recognition Challenge)에서 사람의 인식률을 뛰어넘는 딥러닝 기술과 함께 폭발적으로 발전되고 있다. 이와

같은 발전에도 불구하고 실험실 환경에서 사람의 인지능력을 능가하였어도 최근 자율 주행 교통사고 사례와 같이 실제 환경에서의 발생하는 사고에 불안정성은 해결하기 어려운 난제와도 같다.

기존에 진행되고 있는 딥러닝 기반의 객체 인식 연구 방향으로는 딥러닝 구조 개선을 통한 성능을 향상시키는 방인이 있다. 최근 대표적으로 CVPR에서 발표된 Cascade R-CNN과 같은 사례나 YOLO, YOLOv2, YOLOv3 및 YOLOv4와 같이 YOLO 기존 구조를 활용한 개선 및 업그레이드 형식과 같이 연구되고 있다. 또는 데이터 증식 방안으로 ICCV에서 발표된 CutMix와 Mosaic 등과 같이 기존 데이터셋을 합성하는 방안들의 연구를 통해 물체 인식의 성능 향상을 도모하고 있다[1][2][3][4][5][6].

위와 같이 획기적인 구조와 놀라운 아이디어가 반영된 연구가 지속적으로 이루어지고 있는데, 노이즈와 같은 현상 외에 학습되지 못한 패턴에 대한 주변 환경 인지 기술의 불안정성은 여전히 근본적인 문제점으로 여겨진다. 이는 딥러닝 기술과 같이 데이터 학습 기반의 인공지능 기술의 해결되지 못한 난제이다. 또한 딥러닝 학습 과정에서 요구되는 대규모의 데이터와 연산 컴퓨팅 능력이 요구되는데, 데이터 관리와 이미지 내의 정보를 포함하는 라벨링 작업과 같이 상당한 인적 자원 및 비용이 소모된다. ILSVRC 경진대회의 데이터셋으로 사용되는 ImageNet을 구축하는데 소모된 시간은 4년, 인력으로는 약 4만9천여명이 작업한 것으로 알려져있다. 이러한 사례와 같이 딥러닝의 지도학습에 필요한 데이터 작업의 시간 및 비용 문제는 극복해야할 문제이다.

이러한 문제점을 해결하기 위해 데이터 라벨링 작업을 줄여주는 자기 지도 학습 연구의 수요가 증가하고 있다. 그 뿐만 아니라 예상치 못한 패턴에 대한 이미지 인식 기술의 학습 외 분포 데이터 탐지 능력(Out-of-Distribution Detection)을 높이고자 생성적 대립 신경망(Generative Adversarial Network)와 같은 인지 정확도를 높이는 방안도 새롭게 제시된 후로 지속적으로 성능 향상을 위한 연구가 진행중이다. 다양한 산업 분야에서 안정적인 기술 접목을 위해 공적인 환경을 넘어 상업적인 환경에서의 적용 가능 연구가 진행되어 인력 비용 절감과 예상치 못하는 패턴 데이터 학습을 통해 정확도 향상시킬 필요가 있다[7].

2. 연구 이론 및 내용

1) 관련 연구 이론

이미지 기반의 사물 인식 분야는 크게 Object classification, Object detection, Image segmentation으로 나뉜다. 이미지 내의 물체 클래스를 분류하는 Object classification, 이미지 내의 다중 물체 인지 및 위치까지 표기하는 Object detection, 이미지 내의 픽셀 단위로 객체의 존재를 나누는 Image segmentation

이 있다. 최근 들어 위의 기능을 구현하기 위해 인간과 같이 기계가 스스로 학습하여 판단하는 머신러닝(Machine learning) 기술이 사용되고 있다.

머신러닝의 한 분야로 사람의 뉴런과 같은 특성을 반영한 인공신경망의 발견과 심층 신경망으로 발전을 통해 딥러닝(Deep learning) 기술이 발달되었고, 그 이후에 사람의 시각적 요소를 반영한 CNN(Convolutional Neural Network) 알고리즘은 이미지 기반의 인식 분야에 현재까지 도달한 성능 발전의 토대가 되었다.

딥러닝 기술은 지도 학습(Supervised learning)을 기반으로 학습되어지는데, 정답이 있는 데이터를 활용하여 데이터를 학습시키는 것이다. 이러한 딥러닝 기반의 학습은 대규모의 데이터와 데이터에 맞는 정보를 포함한 Annotation 파일이 요구된다. Annotation파일은 이미지 내에 존재하는 사물의 클래스 및 위치 정보를 담은 파일로, 사람의 수작업으로 통해 생성되어지는데 이를 라벨링 작업으로 부른다.

딥러닝 학습을 위해서는 데이터셋의 구축이 필수적이다. 특히 데이터셋의 양과 질에 따라 인식의 결과에 상당한 영향을 끼치기에 상당한 인력과 시간이 필요하다. 기존의 공유 데이터셋으로 알려진 MS COCO 와 같은 데이터셋은 학습 데이터셋의 양으로만 수십에서 수백만장이고, ImageNet 데이터셋의 학습 데이터셋은 120만개의 이미지를 포함하고 있다. 그리고 대중적으로 많이 이용되는 수십에서 수천개의 카테고리로 구성되어있다. 그러나 공적인 사물을 대상이 아니거나 상업적인 용도로 커스텀한 데이터를 인식하기 위해서는 별도의 데이터셋을 새롭게 구성해야 한다.

안정적인 높은 성능을 위해 대규모의 데이터셋 구축은 선택이 아니라 필수적이다. 인력과 비용 절감을 위해 데이터 증식 분야에 대해서도 연구가 지속되고 있다. 그 중에 하나는 기존의 이미지를 조작(Manipulation)을 통해 가공하여 변형된 이미지를 생성하는 것이다. 이는 기존 이미지를 회전, 반전과 기하학적 변환이나 흐릿한 효과, 선명한 효과, 노이즈 추가 등이 존재한다. 이 외에도 딥러닝 기술을 활용하여 새로운 이미지를 생성하는 생성적 적대 신경망 (GAN, Genera-tive Adversarial Network)도 2014년 발표 이후부터 지속적인 관련 연구가 이루어지고 있다. 이는 기존의 이미지 조작 기반의 생성 데이터에 비해 입력 이미지로부터 의존적이지 않고 편향되지 않는 데이터셋 구축 방안으로 평가되어 지고 있다.

현재 물체 인식(Object detection)에 대한 연구는 대규모 공유 데이터셋 기반으로 CNN으로 이루어진 심층 신경망의 구조 개선을 통해 성능 향상을 도모하고 있다. 물체 인식의 신경망 구조는 크게 두 가지 부류로 나뉘는데, 먼저 선보인 2-stage detector 구조로 물체의 위치 후보군을 추출하는 1-stage와 추출된 각 영역의 Classification과 Box regression을 수행하는 2-stage로 구성되어 있다. 대표적인 알고리즘으로는 R-CNN, Fast R-CNN, Faster R-CNN 등이 알려져 있다. 다른 신경망 구조로는 1-stage로만 이루어진 구조이다. 2-stage 구조와는

다르게 물체 후보군 제안과 Classification을 동시에 수행하게 된다. 대표적으로 YOLO 시리즈와 SSD 알고리즘이 있다[8][9][10].

주로 물체 인식의 평가 방안으로 AP(Average Precision) 수치로 이루어진다. detector가 검출한 결과 중 옳게 검출한 비율을 정밀도(Precision)라 하고, 모든 실제값(Ground truth) 중 올바르게 detector 검출한 비율을 재현율(Recall)이라고 한다. 이 두 지표를 반영하기 위하여 정밀도(Precision)와 재현율(Recall)의 PR곡선을 통해 얻어진 면적을 구하여 수치로 표현한 것이 AP 평가 지표이다.

이동체 및 경량 디바이스 등에서 딥러닝 적용을 위한 관련 연구 또한 활발히 진행되고 있다. 이미 스마트폰과 같이 온 디바이스의 인공지능 이미지 인식은 탑재되어 상용화가 되었으며 발전하고 있다. 또한 상대적으로 안정성과 높은 성능이 요구되는 환경에 실시간성을 보장하기 위한 하드웨어 구조 개선 및 GPU 연산 가속화 기술에 대한 연구도 꾸준한 수요가 지속되고 있다.

2) 연구 진행 내용

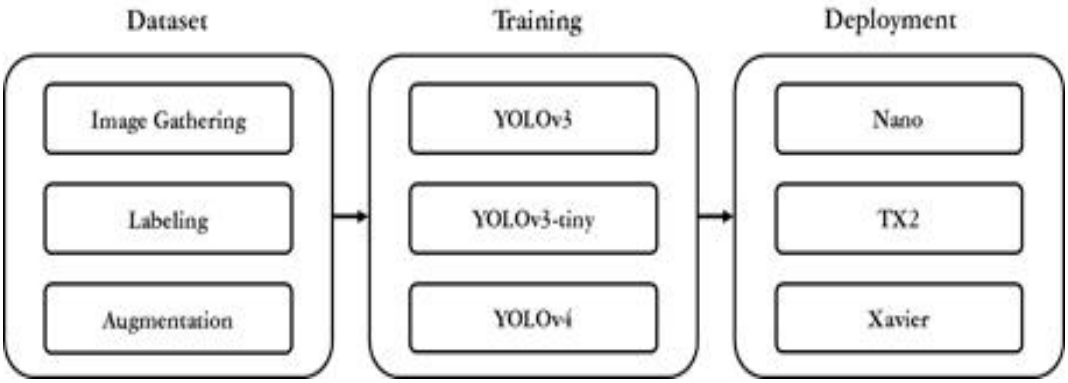
본 논문에서는 특화된 환경에서 Object detection의 성능 향상을 위한 데이터셋 구축 프레임 워크를 제시한다. 이를 검증하기 위해 딥러닝 학습을 통해 성능 평가를 실시하고 실제 환경에서 테스트를 통해 정확도를 검증한다.

먼저 사물 인식(Object detection)을 위한 딥러닝 학습에 필요한 커스텀한 데이터셋을 구축하였다. 먼저 공유 데이터셋에서 얻지 못하는 커스텀 데이터셋을 대상으로 온라인 상의 데이터 수집을 실시하였고, 부족한 양과 편향된 데이터를 보완하기 위해 데이터 조작(Manipulation)과 딥러닝 GAN(Generative Adversarial Network)으로 이루어진 프로세스를 구축하여 데이터 증식을 실시하였다. 해당 데이터셋 기반으로 성능 검증을 위해 1-stage 객체인식 알고리즘인 YOLO 구조로 학습을 실시하였다. 이는 정확도와 실시간 인식시간의 절충된 알고리즘으로 빠른 시간 내에 적절한 정확도 결과를 가져다주기에 이를 적용하였다. 학습을 통해 최적 모델을 생성하였으며 성능 검증을 위해 AP 평가지표를 활용하였다. 이를 Test dataset 으로 mAP(@0.5)로 기준으로 기존 데이터셋 대비 평균 10~ 20% 의 성능 향상과 최대 36% 성능 향상을 확인하였다. 각각의 증식 기법에 의해 증식된 이미지 개수 당 정확도 향상 비율을 분석하였다. 그 결과로 기존 Manipulation 기반 생성된 이미지 개수 당 정확도 향상 비율은 GAN 기반의 생성된 이미지 정확도 향상 비율에 상당히 떨어짐을 확인할 수 있었다.

추가적으로 생성된 최적 모델을 이용하여 실제 환경에서 임베디드 기반 물체 인식 성능 테스트를 진행하였다. 이 때 ROS 기반의 UGV(Unmanned Ground Vehicle)의 플랫폼을 활용하여 물체 인식 성능을 확인하였다. 또한 UGV의 탑재된 임베디드 보드인 Nano, TX2, Xavier에 인지 속도 성능을 확인하였다. YOLOv3-tiny 기준으로 Xavier기반 평균 54FPS, TX2 기반 평균 24FPS, Nano 기반 평균 12FPS의 속도를 제공하였다. YOLOv3기준으로 Xavier기반 평균 14FPS, TX2 기반 평균 8FPS, NANO 기반 평균 3FPS의 속도를 제공하였다.

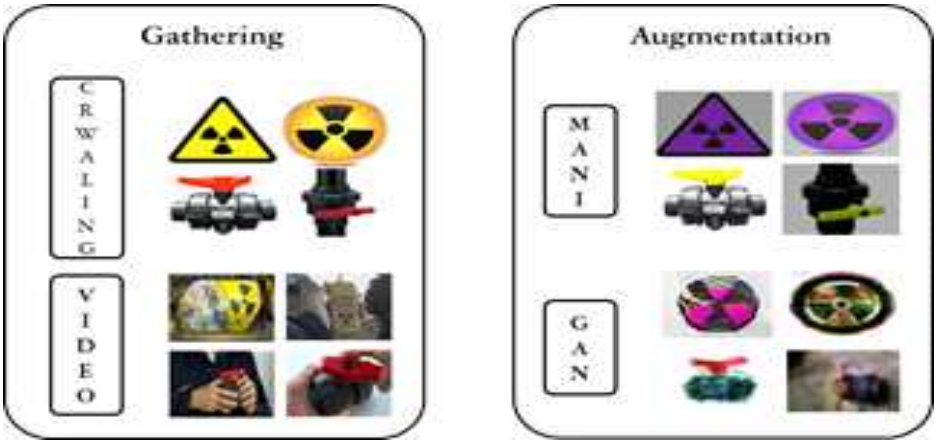
II. 데이터 증식 프레임 워크

객체 인식을 진행하기 위해 딥러닝 데이터셋 구축부터 알고리즘 학습을 통한 모델 생성, 임베디드 기반 카메라 이미지의 적용의 순으로 다음 그림 1과 같이 진행된다. 먼저 학습을 위해서는 인지 대상의 데이터셋이 요구된다. 기존의 KITTI, MS COCO, ImageNet 등과 같이 정형화된 공유 데이터셋이 존재한다. 해당 데이터셋들은 다양한 카테고리를 포함한 수십에서 수백만장의 데이터로 구성되어 있다. 그러나 일반적으로 차량, 개, 고양이, 컵 등의 종류로 구성되어 있기 때문에 대중적이지 않고 상업적인 타겟을 대상으로 객체 인식을 하고자 한다면, 별도로 커스텀 데이터셋을 직접 구축하는 과정이 필요하다. 데이터 수집, 데이터 가공을 통한 데이터 증식 및 타겟의 클래스와 위치 정보를 담는 Annotation 파일 생성을 실시하여 데이터셋을 구축한다. 그 다음으로 생성되어진 데이터셋을 이용하여 객체 인식 알고리즘 기반으로 딥러닝 학습을 실시하여 최적 Detector 모델을 생성한다. 생성된 Detector 모델을 PC 에서나 임베디드 보드 시스템에 포팅하여 객체 인식 시스템이 적용된다.



<그림 1> 객체 인식 프레임 워크

객체 인식 분야에서는 이미지내의 정확도를 높이기 위해서는 방대한 이미지 양이 필수적이고, 또한 편향되지 않는 다양한 데이터가 필요하다. 기존의 커스텀 한 데이터셋의 한계점으로는 시간과 비용이 제한적이기에 데이터의 많은 양을 확보하기에는 어려움이 있다. 또한 Object detection인 경우에는 이미지내의 타겟의 위치를 포함하는 Annotation 파일을 생성하는 라벨링 작업이 필요하다. 이러한 현실적인 제약 조건에서의 효율을 높이기 위해, 이 논문에서는 취득하기에 제한적인 커스텀 데이터를 수집을 실시하였고, 목표물의 위치 정보를 포함하는 라벨 작업 생성과 같은 Data preprocessing을 실시하고, 취득 데이터를 활용하여 데이터 가공 및 딥러닝 기반의 이미지 재생성과 새로운 이미지 데이터를 생성하여 증식 데이터셋 구축 방안에 대해 초점을 두었다.



<그림 2> 데이터 수집 및 증식 방안

1. 데이터 수집 및 전처리

온라인상의 데이터를 수집하는 하나의 방법으로 데이터 크롤링 기법이 있다. 이는 Google과 Bing과 같이 인터넷상에서 일반 사용자의 접근이 용이한 곳에서 얻을 수 있는 이미지 데이터들을 프로그래밍 언어 기반의 코드를 활용하여 정형화된 형태(HTML)의 규칙을 분석하여 간편히 원하는 데이터를 추출 가능하도록 하는 알고리즘이다. 본 논문에서는 Python 기반의 Icrawler를 활용하여 방사능 표시판과 밸브의 이미지를 아래의 그림 3과 같이 수집하였다. 추가적으로 Youtube와 같은 영상 플랫폼에서 영상 추출 및 이미지 변환을 통하여 데이터를 취득하였다.

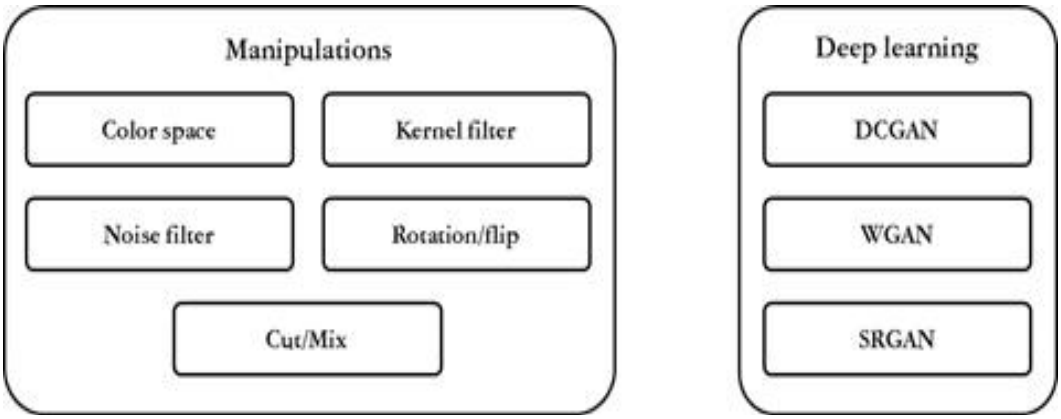
<그림 3> 웹 크롤링 기반 수집된 이미지

그러나 이러한 데이터는 관련성 없는 데이터, 많이 훼손되거나 변형되어 정보를 판단할 수 없는 데이터 그리고 일반적으로 해당 데이터 크기나 모양이 다양하지 못한 점이 있다. 이 과정에서 크롤링된 이미지를 선별하는 작업을 수행하

였다. 또한 얻어진 해당 이미지 내의 목표물의 위치를 표기하는 라벨링 작업을 실시하였다. 이와 같이 데이터 수집 및 전처리 작업을 완료 후, 앞서 언급한 내용과 같이 데이터 개수 부족, 편향된 데이터를 해결하기 위해 수집되어진 데이터셋을 기반으로 가공하여 생성하도록 한다.

2. 데이터 증식

크롤링 및 영상을 통한 데이터 수집을 통해 데이터셋을 구축하기에는 충분하지 않다. 데이터셋 증식을 위한 연구는 크게 두 가지의 분류로 나뉘어 연구가 진행되고 있다. 기존 이미지를 가공하는 방식으로 색깔 변환, 회전, 노이즈, 상하 또는 수직 반전, 블러 변환 등의 Image manipulation이 있고, 최근 딥러닝 연구의 트렌드인 GAN 의 데이터 증식이 있다. 2014년 학회에서 Yan Goodfellow 저자에 의해 발표된 GAN(Generative Adversarial Net)은 비지도 학습 부분에서 현재까지도 연구가 지속되고 있다. Generator 와 Discriminator의 경쟁적인 두개의 모델을 제시해 서로의 성능을 점점 개선해나가는 것이 주요 개념이다.



<그림 4> 데이터 증식 방안 분류

따라서 우리는 제한적인 커스텀한 객체 인식의 데이터셋 구축 한계점을 해결하기 위해 이미지 Manipulation과 딥러닝 기반의 GAN 알고리즘을 포함한 이미지증식의 프레임 워크를 제시한다. 증식 데이터셋을 구축하고 기존 수집된 데이터셋과 비교를 통해 알고리즘 성능을 검증하였다.

1) Image Manipulation

본 논문에서는 날씨, 각도, 영상의 퀄리티 및 카메라의 움직임 등에 의해 변형

되는 이미지를 반영하기 위해 Image manipulation 기능의 코드를 설계하였다. 그 중 하나인 이미지 회전과 반전은 타겟의 조건이나 이동체의 움직임 또는 지상의 경사에 따른 카메라 앵글의 변화로 회전되어진 타겟의 정보를 반영할 수 있는 데이터를 반영하고자 한다. 이와 같이 커널 필터링과 랜덤 노이즈를 통해 다양한 환경에서의 적응을 높이기 위해 노이즈를 적용하여 외부적인 요인을 반영할 수 있는 효과를 얻고자 하였다. 또한 타겟의 특성에 따라 다양한 색감을 반영하여 인식하기 위해 색깔 변환을 하고자 하였다.

<표 1> Manipulation의 세부 기능 및 효과

Manipulation	효과	환경 모사
Geometric transform	회전 및 반전	도로 지형, 이동체 움직임 따른 Orientation 변화 반영
Kernel filtering	흐릿한 효과	카메라 및 응답속도 조건 반영
Random noise	가우시안 노이즈	날씨 및 물체 환경 조건 반영
Color transform	물체 색 변환	다양한 물체 색감 반영

따라서 각각의 기능을 적용하는 방식에 대해 위의 표1과 같이 적용하였다. 또한 기존의 이미지의 목표물 좌표값이 포함된 Annotation 파일 또한 재생성되어야 한다. 각각의 기능에 따라 변화되는 좌표값 또한 반영되어 생성하는 것이 필요하다. 이에 따라 변화된 목표물 픽셀 위치 정보를 계산하여 Annotation 파일을 생성하는 자동화 라벨링 알고리즘을 설계하였다. 세부적인 알고리즘의 프로세스는 아래의 표2와 같다. 먼저 각 증식 데이터셋 생성을 위한 Operation 들의 Probability를 정의하고, 기존 이미지들의 Path를 List로 저장하여 하나씩 이미지의 위치를 가져온다. 그리고 균일한 랜덤 변수를 생성하여 Operation의 확률들과 비교하여 낮을 때, Operation이 적용된 이미지를 생성하고, 변형된 이미지에 맞게 변화된 목표물 픽셀 위치 정보를 담은 Annotation 파일을 저장하였다. 각각의 Operation에 따라 해당 물체의 변화를 반영하기 위해 다음으로 물체 위치 관련 수식을 설명한다.

<표 2> 이미지 증식 및 annotation 생성 프로세스

order	process
1	Initialize parameters of augment operation probability
2	for path in train dataset path list
3	for operation in augment operations
4	Set a random variable α in uniform number(0,1)
5	If operation.probability > α
6	Create augment image as operation
7	Calculate changed target pixel location(x,y) as operation
8	Save annotation file applying new target pixel location

Rotation은 기존의 이미지의 중심점을 기준으로 회전을 시켜 이미지를 생성하였다. 특히 회전 각도는 -15도 ~ 15도 사이로 제한을 두었으며 이때 출력 이미지는 기존의 이미지 크기와 동일하게 하고 회전에 따른 여백의 공간은 비워두었다. 다음 식과 같이 목표물 픽셀 위치를 계산하였다. 먼저, 라벨 파일의 x, y 위치는 1로 Normalization 되어 표기된다. 이미지 중심에서 기존 이미지 내의 타겟 사이의 거리를 r로 설정하고, 는 중심으로부터 목표물까지 이은 선과 x축 사이의 각도로 정하였다.

데이터셋의 이미지들은 타겟이 중심에만 위치하지 않고 다양한 크기와 위치에 놓여 있을 수 있기 때문에 Annotation의 정보도 동일하게 회전되어야 한다. 랜덤 회전 각도에 따라 Annotation파일 내용도 같은 식1, 2로 변화된 타겟의 픽셀 위치를 구하여 Annotation 파일을 저장하였다. θ는 기존 이미지 내에 중심 기준의 물체 중심 좌표간의 각도이며, γ는 랜덤 회전 각도이다.

$$r = (|x-0.5|)^2 + (|y-0.5|)^2,\ \theta = \begin{cases} \tan^{-1}(\frac{0.5-y}{x-0.5}) \text{ if } y < 0.5 \\ 180 + \tan^{-1}(\left|\frac{0.5-y}{x-0.5}\right|) \text{ if } y > 0.5 \end{cases} \quad (1)$$

$$target\ pixel\ location = \begin{cases} X + r\cos(\theta-\gamma) \\ Y - r\cos(\theta-\gamma) \end{cases} \quad (2)$$

<그림 5> 회전 이미지 생성 모습

(상단 : 수집 이미지, 하단 : 랜덤 회전 이미지)

Flip은 이미지의 상하 또는 좌우 반전된 이미지를 생성하는 것이다. 해당하는 라벨링 작업을 실시하여야 되는데, Horizontal flip과 Vertical flip에 따라 변화되는 목표물 위치값을 아래와 같은 식 3과 같이 타겟의 위치는 이동할 것이다. 따라서 반전 모드에 따라 Annotation 파일은 아래의 식과 같이 생성하였다. 이 때, 픽셀좌표가 1로 Normalize된 좌표값이므로 아래와 같다.

$$target\ pixel\ location = \begin{cases} 1-X \\ Y \end{cases},\ target\ pixel\ location = \begin{cases} X \\ 1-Y \end{cases} \quad (3)$$

<그림 6> 반전 이미지 생성 모습
(상단 : 수집 이미지, 하단 : 반전 이미지)

Kernel filtering은 이미지를 Blurring, Sharpening, Embossing 및 Edge detection 등에 활용되는 기능으로, 우리는 흐릿한 이미지를 데이터셋에 포함하

여 이동체의 움직임 및 이미지의 품질의 영향을 덜 받고자하였다. 이 때 균일 필터는 11x11 픽셀 크기로, 입력 이미지가 Kernel을 통과하여 이미지를 생성하였다. 그 이상의 크기 필터는 객체의 특성을 없앨 수 있는 가능성이 농후하여 사용하지 않았다.

<그림 7> 흐릿한 이미지 생성 모습
(상단 : 수집 이미지, 하단 : blur 적용 이미지)

Random noise은 각 이미지의 픽셀에 임의의 값을 더하여 특수한 효과를 주기 위해 사용된다. 전체적인 픽셀값에 일괄적으로 임의값을 더하는 Uniform noise와 정해진 범위내의 랜덤값을 더하는 Gaussian noise, 픽셀값에 0 또는 1의 갑을 주입하여 희고 검은 노이즈를 더하는 Salt&Pepper등이 있다. 우리는 S&P 모드와 Gaussian을 동일한 비율로 이미지를 생성하였다. Seed는 0.4로 설정하였다. 아래의 그림 8은 랜덤 노이즈가 적용된 예로, 해당 이미지의 하단은 Gaussian 및 S&P가 적용된 이미지이다.

<그림 8> 랜덤 노이즈 이미지 생성 모습

(상단 : 수집 이미지, 하단 : 노이즈 적용 이미지)

Color space는 우리가 보는 색을 3차원 공간에 표현하는 것으로, 색의 속성인 명도, 채도, 색상을 이용하여 모든 색을 표현한다. 이러한 특성을 이용하여 이미지의 Color space를 임의로 조절하여 이미지 변환을 실시하였다. 특히 밝기, 대비, 채도 및 색조와 같은 특징들을 변환함으로써 다양한 환경 조건에서 뿐만 아니라 타겟의 다양한 색깔에도 강건한 데이터셋을 구축을 가능케 한다. 밝기와 대비, 채도, 색조의 Random factor들은 0.4로 설정하였다. 균일한 Factor 값 [max(, 1-factor), 1+ factor] 범위에 랜덤하게 적용되어 이미지에 효과가 적용된다. Color space transform 변환의 한 예로 아래의 그림 9와 같다. 왼쪽 이미지는 수집된 원본 이미지로, 각 특성에 Random seed를 주입하여 중간과 오른쪽 이미지와 같이 변환하였다.

<그림 9> Color space Transform 이미지 생성 모습

(왼쪽 : 수집 이미지, 중간/오른쪽 : 적용 이미지)

Rotation, flip을 제외한 Operations(Color space, Blur, Random noise)은 이미지를 재생성하여도 목표물 위치와 크기는 변화하지 않으므로, 기존의 Annotation 파일의 데이터를 그대로 읽어 들여 라벨링 작업을 실시하였다. 위의 이미지 가공을 위한 파라미터 설정은 표3과 같이 선택하였다. 이는 미흡하거나 과도한 필터를 통해 사물을 분별하지 못할 것을 방지하기 위하여 실험을 통해 정하였다.

<표 3> Manipulation의 operation 파라미터

Parameter Name	Value (Option)
operation probability	1
Rotation limit	-15 ~ 15 deg
Random noise mode	Gaussian, S&P
Random noise seed	0.4
Blur filter	11 x 11 pixel
Brightness, Contrast, Saturation, Hue factor	0.4

2) GAN 기반의 이미지 증식

위와 같이 온라인 크롤링과 비디오 이미지에서 취득한 기존의 데이터셋은 몇 가지의 한계점이 존재한다. 특히 객체의 특성에 따라 또는 제작업체에 따라 색깔, 형태 등이 천차만별이다. 예를 들어 방사능 표시판과 같은 심볼은 국제적으로 표준으로 이용되기에 모양의 특징이 특별하고 고정적인 부분과 변화되는 부분이 뚜렷하다. 그와 다르게 밸브의 경우에는 제작하는 기업에 따라 천차만별이고 모양이 고정하지 않고 색깔마저 다양하게 존재한다. 심지어 기업에서 제공하는 데이터가 적어 데이터셋 구축하기가 매우 어렵다. 이를 보완하면서 편향된 데이터셋을 해결하기 위해 딥러닝 기반의 GAN의 이미지 증식을 실시하였다.

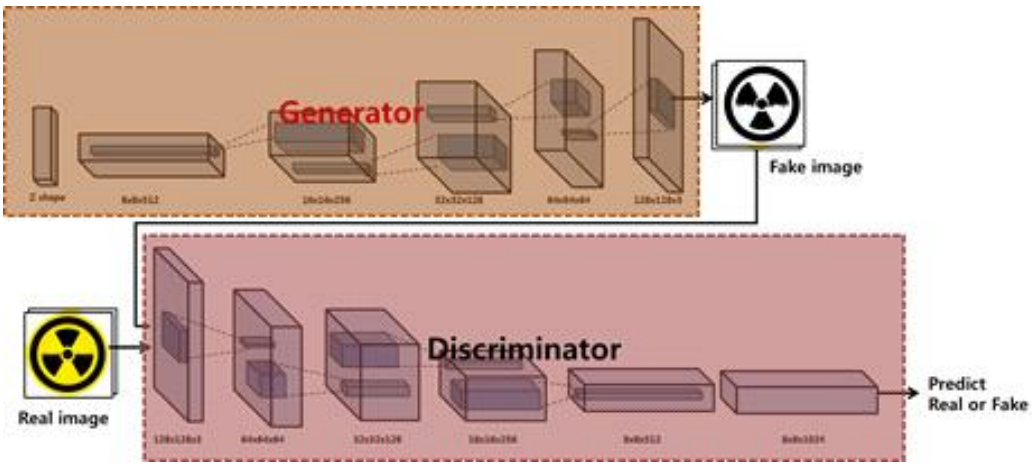
GAN(Generative Adversarial Networks)은 2014년도에 새롭게 발표되어 머신러닝 분야에서 새롭게 등장한 아이디어로, 이미지를 만들어내는 Generator와 적대적인(Adversarial) 위치에서 Generator의 성능을 평가하는 Discriminator가 존재한다. 학습을 통해 서로의 성능을 점차 개선하여 실제 이미지와 새롭게 생성한 가짜 이미지를 구별할 수 없는 확률(0.5)에 도달하는 것을 목표로 설계된 모델이다.

Discriminator 입장에서는 기존 이미지 Sample x는 D(x) = 1이 되도록 하고, Generator에 임의의 Noise distribution으로부터 뽑은 Input z를 놓고 만들어진 Sample에 대해서는 D(G(z))=0이 되도록 학습한다. 이는 Minimax problem이라 할 수 있으며, 이를 수식으로 정리하면 다음과 같이 표현이 가능하다.

① DCGAN[11]

초기에 발표된 GAN 학습 시 안정성이 떨어지는 심각한 문제가 존재하였으며, GAN의 결과물인 새롭게 만들어진 Sample은 정량적 척도가 없기에 모델의 성능

평가가 상당히 어려운 점이 있다. 이후에 2016년에 발표된 DCGAN(Deep Convolutional GAN)은 기존 GAN이 Fully-connected Neural Network를 구성되어 있는 것과 달리 CNN 구조를 적용하였다. 이외에도 안정적인 학습 모델을 위해 다음과 같은 구조를 채택하였다. 기존 Discriminator에서 적용된 Pooling layers를 Strided convolutions로 바꾸고, Generator에서 Fractional-strided convolution을 사용하여 Feature map의 사이즈를 확장하였다. 또한 Fully connected hidden layer를 제거하고 Generator 활성함수를 Tanh 와 ReLU를 사용하였다. Discriminator는 LeakyReLU를 사용하였다. 다양한 구조에 대한 실험 끝에 최적화된 구조를 발견하였다. 이후에도 이 구조를 바탕으로 후속 GAN 논문들이 발표되고 있다. 이뿐만 아니라 GAN의 Input noise z에 대한 의미를 테스트해보았다.



<그림 10> DCGAN 구조

수학적 근거 하에 추론을 한 것은 아니나, 실험을 통하여 최적의 구조를 생성한 것을 알 수 있었다. 따라서 우리는 커스텀한 데이터셋의 최적화된 파라미터 설정을 반복 학습을 통해 비교 분석하였고 다음과 같이 설정하였다. Epoch는 1000, Batch size는 64, Learning rate of generator는 0.00004, Learning rate of Discriminator는 0.0004, Adam initial decay 는 0.5, 0.999, Image size는 128x128로 하였다.

<그림 11> DCGAN 기반 생성 이미지

② WGAN[12]

비지도 학습은 답이 있는 데이터셋이 아닌 주어지는 데이터의 분포를 학습을 목표로 하게 된다. Latent variable z의 분포를 가정하여 주입하면서 Discriminator와 Generator간의 관계를 학습을 진행한다. 학습에 따라 입력 이미지의 분포에 가깝게 도달하도록 한다.

그러나 GAN은 Discriminator와 Generator간의 균형을 유지하기 어려우면서, 학습이 완료된 이후에도 Mode dropping이 발생하는 오류가 존재한다. 주요한 원인은 Discriminator가 충분히 감별하는 역할을 해주지 못하기에 모델이 최적점까지 학습하지 못하였기 때문이다. 이러한 문제점을 해결하기 위해 Wasserstein GAN은 크게 아래의 두 가지 방안을 제시한다.

Discriminator는 진실 거짓을 분별하기 위해 Sigmoid를 사용하여 결과물은 예측확률 값을 갖는다. 대신 새로 정의한 Critic을 사용하여 EM distance로 확률 분포 간의 거리를 측정하여 학습을 실시하는 것이다. 이는 기존에 사용하는 JS(Jensen-Shannon), KL(Kullback-Leibler) divergence는 매우 Strict하게 거리를 측정하여 Continuous하지 않은 경우가 종종 발생한다. 따라서 EM(Earth-Mover) distance는 보다 Gradient를 잘 전달시키고 Critic과 Generator를 최적점까지 학습을 가능케 하였다.

EM distance는 두 확률 분포의 결합 확률 분포 중에서 x, y의 거리의 기댓값을 가장 작게 추정하는 것을 목적으로 둔다. Earth Mover's distance는 명칭을 보듯이 모래와 같이 더미를 옮기는 데에 드는 비용을 뜻한다. 이와 같이 한 분포에서 다른 분포의 모양으로 옮겨지는 더미의 최소 비용을 의미한다. 이는 기존의 JS divergence 방식에 비해 Continuous한 상황에서 선형적 특성의 결과를 도출할 수 있기에 학습을 보다 안정적으로 진행할 수 있다. EM distance는 아래의 식 4와 같이 두 확률 분포의 결합 확률 본포 $\Pi(P_r, P_g)$ 중에서 x, y 간의 기

댓값을 가장 작게 추정하도록 한다.

$$W(P_r, P_g) = \inf_{\gamma \to \Pi(P_r, P_g)} E_{(x,y)} [\, \| x - y \| \,] \tag{4}$$

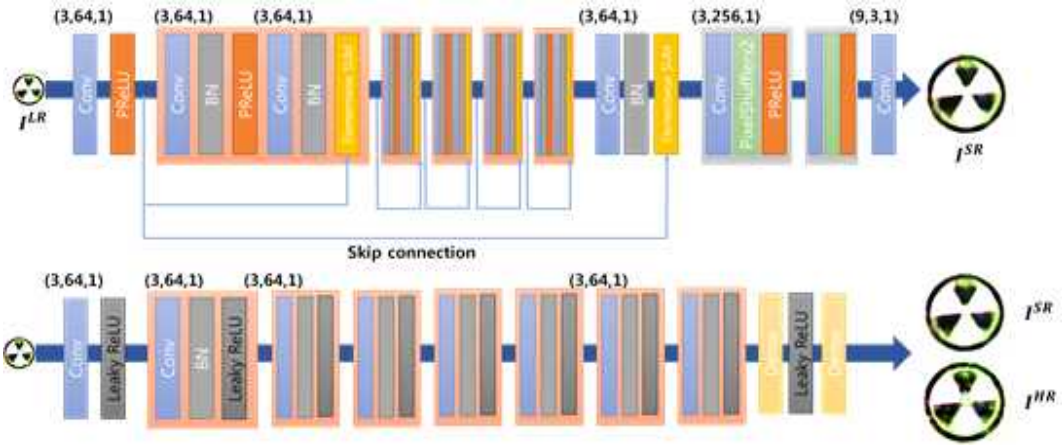
우리는 부족한 데이터셋의 양을 보완하고자 DCGAN(Deep Convolutional GAN), WGAN(Wasserstein GAN) 등의 네트워크를 구성하였다. DCGAN은 기존 GAN에 CNN를 접목하여 성능을 향상시켰고, WGAN은 기존 Discriminator와 Generator의 균형을 유지하면서 학습의 어려움을 해결하고자 Discriminator의 새로운 Critic을 사용하고, Sigmoid를 적용하였다. 또한 EM이 Distance를 제시하여 확률 분포간의 거리를 측정하는 기존의 KL divergence를 대체하였다. 이 두 개의 네트워크를 바탕으로 Batch size는 64, epoch 는 최대 5000, Learning ratio of Discriminator 는 0.0004 , Learning rate of Generator는 0.0004, Momentum of stochastic gradient descent(beta) 는 0.5 등으로 학습 환경 및 반복 테스트를 통해 학습을 진행하고 이미지를 생성하였다.

<그림 12> WGAN 기반 생성 이미지

③ SRGAN[13]

위의 DCGAN, WGAN를 적용하여 이미지 증식을 실시하였다. 그러나 위의 알고리즘을 적용했을 때, 고정적인 입력 이미지를 바탕으로 동일한 크기의 증식 이미지를 생성하였다. 이는 DCGAN는 128 x 128의 이미지를 생성하였고 WGAN는 256 x 256 의 이미지를 생성하였다. 물론 고해상도의 이미지로 생성은 가능하지만, 학습하는 이미지의 크기가 고해상도 이미지가 제공되어야 하며 생성하는 이미지도 커짐에 따라 소요되는 시간이 오래 걸린다. 이와 같은 상황에서 객체 인식의 알고리즘에 고해상도의 증식 이미지를 추가적으로 제공을 목적

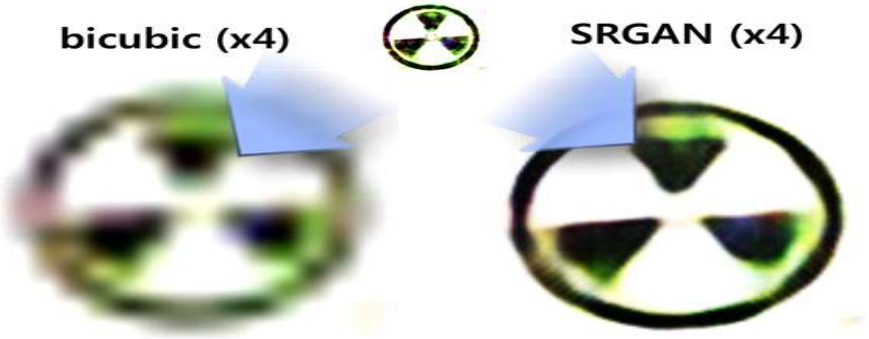
으로 하여 SRGAN을 이용하였다. SRGAN은 저해상도인 이미지를 고해상도의 이미지로 변환하는 SR(Super-Resolution)을 목적으로 생성되었다. 또한 기존의 SR 연구에 문제점인 High frequency detail들이 부족한 점을 보완하고자 하였다.



<그림 13> SRGAN 구조

SRGAN 구조는 딥러닝 구조 발전 흐름과 함께 Skip connection 구조를 채택하여 Residual blocks들로 이루어져 있으며 Leaky ReLU와 같은 최적화된 활성화 함수를 적용하였다.

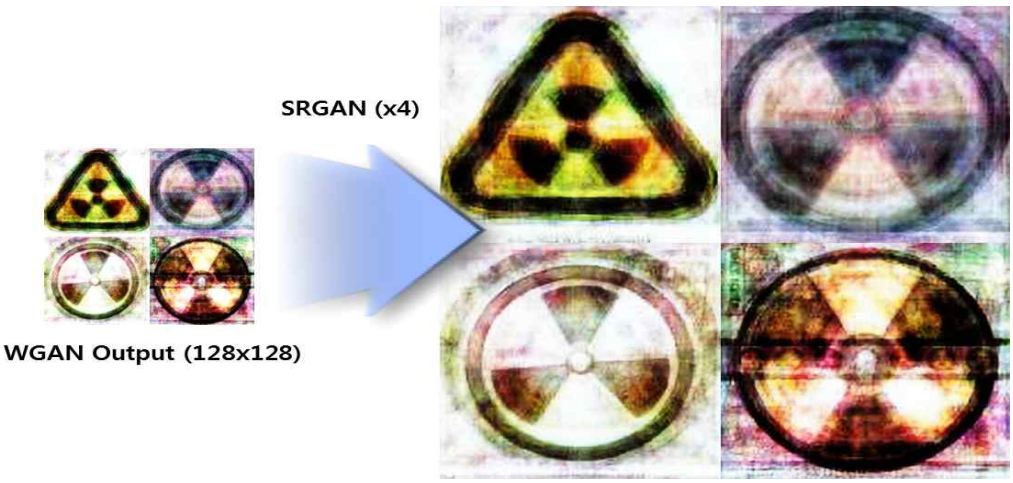
주로 SR의 수행할 때 평가 인자로 여기는 PSNR을 높이고자 MSE를 최소화시키고자 한다. 그러나 PSNR이 높다고 하여 세세하게 질감이 살아있는 고해상도의 이미지를 생성하였다고 보기 어렵다. 이는 인접 픽셀간을 활용하여 픽셀 크기를 확장하는 Bicubic 방식과 비교하며 평가하면 보기 쉽다. 아래의 그림 14는 학습 시에 확인한 SR 이미지와 Bicubic 이미지를 비교한 것이다. 왼쪽 이미지는 128x128 픽셀 크기의 원본 이미지이다. 나머지 이미지는 4배의 해상도 높여 생성한 이미지이다. 중간 이미지는 Bicubic 기법을 통해 생성하였고, 오른쪽 이미지는 SRGAN을 통해 생성한 이미지이다.



<그림 14> 4배 고해상도 Super-Resolution 생성 예

(왼쪽 : 원본 이미지, 중간 : Bicubic 생성 이미지, 오른쪽 : SRGAN 생성 이미지)

본 논문에서는 DCGAN, WGAN로 증식된 데이터셋을 입력으로 주입하여 4배의 해상도 크기를 확장하여 고해상도의 생성할 수 있도록 하였다. 이는 DCGAN과 WGAN의 출력 이미지인 128x128 저해상도의 이미지를 복원하여 데이터셋으로 추가한 것을 의미한다. 이를 통해 객체 인식 딥러닝 학습 정확도 향상을 도모하였다. 아래의 그림 15는 학습된 SRGAN을 통해 생성한 이미지이다. 단지 픽셀의 확장을 넘어 이미지의 Detail을 이질감이 느껴지지 않도록 고해상도의 이미지를 생성한 것을 확인할 수 있다.

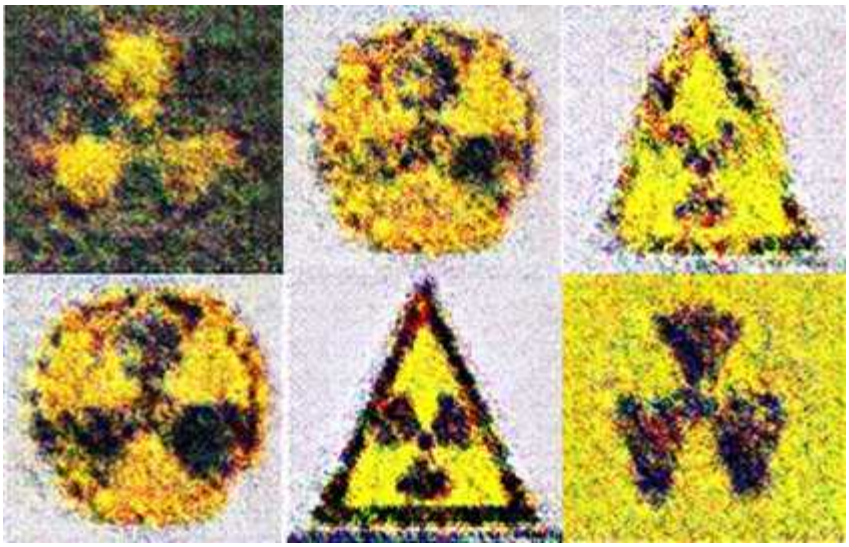


<그림 15> SRGAN 기반 생성 이미지

3. 증식 데이터셋 구축

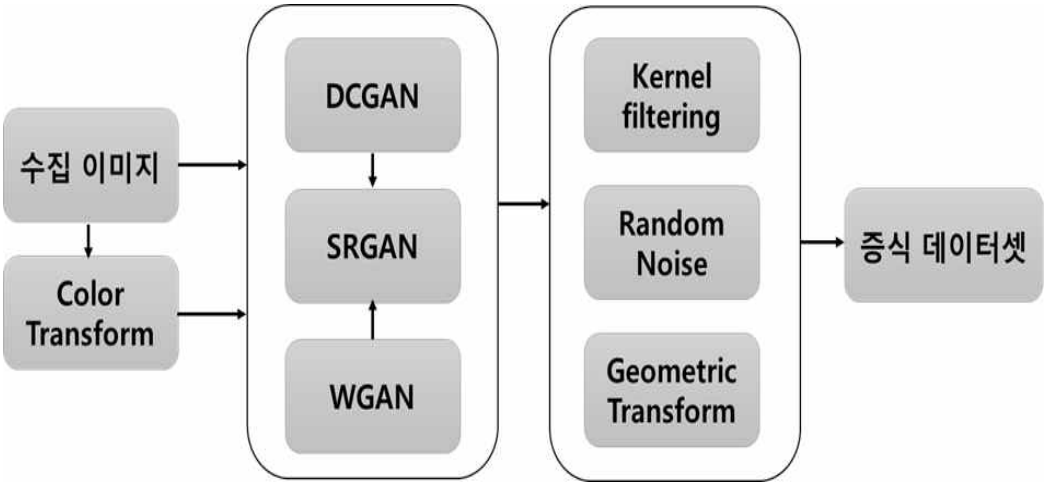
GAN 기반으로 이미지 증식을 위해서는 생성하고자 하는 목표물의 이미지를

다양하게 입력해주어야 한다. 입력 이미지를 일관된 색을 지닌 이미지를 재생성하는 문제점이 발생한다. 기존 웹 크롤링 또는 영상을 통해 수집된 이미지만을 이용하여 DCGAN 기반의 생성된 이미지는 아래의 그림 16과 같다. 실제 수집된 이미지는 노란색으로 구성된 목표물이기에 증식되어지는 목표물도 주로 노란색으로 이루어져 있다.



<그림 16> 수집된 이미지 기반의 편향된 DCGAN 생성 이미지

이와 같은 현상을 보완하고자 수집된 이미지를 기반으로 Manipulations의 Color transform를 적용하여 다양한 색으로 변경된 타겟이 포함한 이미지를 생성하였다. Color space 이미지와 수집이미지를 DCGAN, WGAN 학습을 위해 입력이미지로 주입하였다. DCGAN과 WGAN 학습으로 생성된 이미지를 데이터셋에 포함하고, 추가적으로 DCGAN과 WGAN로 생성된 증식 이미지를 고해상도 이미지 복원을 실시하기 위해 SRGAN의 입력 이미지로 학습하여 고해상도의 이미지를 생성하였다. 그렇게 GAN 알고리즘에 의해 생겨난 증식 이미지, Color space 이미지, 수집이미지를 Manipulations의 Kernel filtering을 적용하여 흐릿한 이미지와, Gaussian noise를 첨가한 이미지, Geometric transform을 적용한 회전 및 반전된 이미지를 생성하였다. 이와 같이 데이터셋 생성 플로우는 아래의 그림 17과 동일하다.



<그림 17> 데이터 증식 프레임워크

웹 크롤링과 영상을 통한 이미지 추출을 통해 수집된 이미지는 11976개, Color transform을 통해 얻은 이미지는 5208개, DCGAN을 통해 증식 이미지는 1063개, WGAN을 통해 증식 이미지는 1390개, SRGAN을 통해 생성된 이미지는 2453개, Image manipulation을 통해 93568개를 생성하여 데이터셋 총 크기는 110450개를 구축하였다.

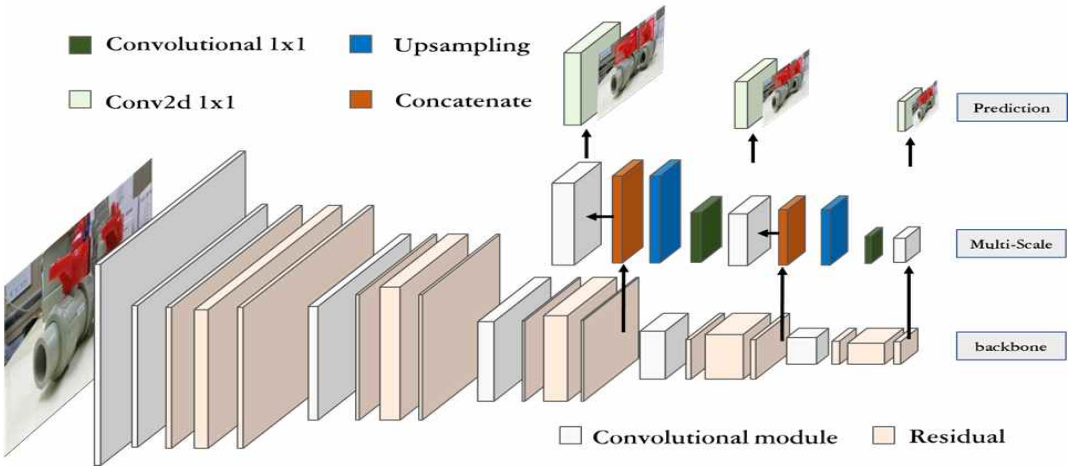
III. 딥러닝 기반의 물체 인식 학습 및 평가

1. 딥러닝 알고리즘

구축되어진 데이터셋을 기반으로 딥러닝 학습을 위해 용도에 따라 알고리즘 선택이 되어져야 한다. 특히 객체 인식의 알고리즘으로는 두 가지인데, 1-stage 알고리즘과 2-stage 알고리즘으로 분류된다. 1-stage의 대표적인 알고리즘은 YOLO, SSD, RetinaNet, EfficientNet 등이 있고, 2-stage 알고리즘으로는 R-CNN, fast R-CNN, Mask R-FCNN, RepPoints 등이 대표적이다. 특히 RCNN은 이미지로부터 수천개에 이르는 박스 위치를 제안하는 stage와 이를 기반으로 분류하는 stage인 분류기로 물체를 검출하는 형태이다. 그러나 높은 정확도에 비해 방대한 연산량에 의해 많은 시간이 소요되는 단점이 존재한다. 그렇기 때문에 로봇, 차량 등에 빠른 응답을 요구하는 시스템 내에서는 1-stage 알고리즘을 기반으로 주로 적용된다. 이는 사용자의 합리적인 요구 선에서 부합하는 정확도를 가지고 빠른 응답이 가능하기 때문이다. 그렇기 때문에 실시간과 같은 응답을 보장하고 높은 성능을 추론하는 알고리즘 연구가 지속되고 있다. 그 중에서 매번 속도 및 정확도를 개선하여 정확도 및 속도의 트레이드 오프 관계에서 좋은 성과를 가지고 릴리즈되고 있는 YOLO 계열의 알고리즘 기반으로 우리의 로봇에 적용하기로 하였다[14][15][16][17][18].

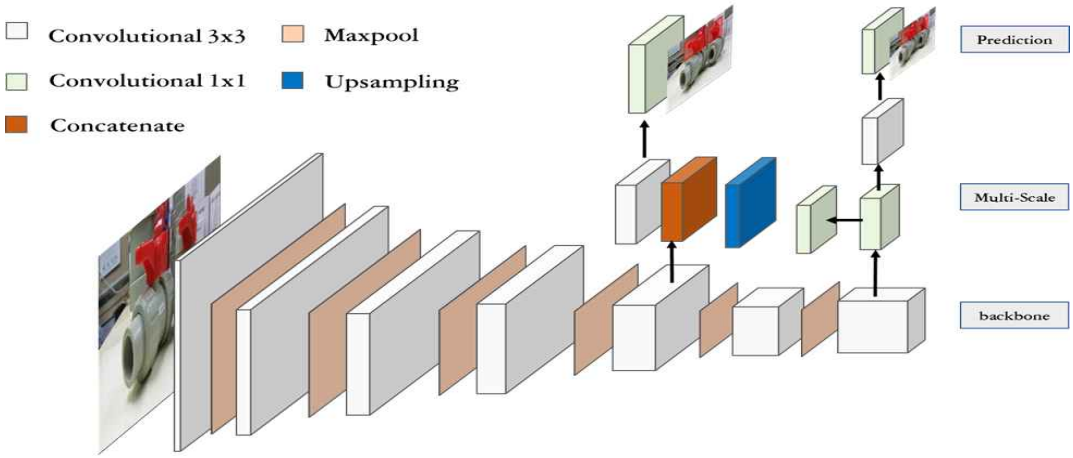
1) YOLOv3 & YOLOv3-tiny

기존의 2-stage 디텍더들의 느린 인지 속도 문제점을 보완하고자 YOLO는 1-stage 알고리즘의 선구자이다. 그 이후에 지속적인 연구 결과로, YOLOv3는 Feature 추출을 위해 사용되는 Backbone의 깊이를 이전보다 증가시키면서 다른 디텍더들에 비해 효율적인 계층을 구성하면서 상대적으로 높은 정확도 및 빠른 속도의 성과를 얻었다. 아래의 그림 18을 보면 이전에는 19개의 계층으로 이루어진 Backbone으로 구성되었다면 YOLOv3는 53개의 계층 구조를 통해 구성된 것을 볼 수 있다. 또한 기존의 Convolution layer와 Max-pool 구조에서 Convolution layer와 Shortcut 구조를 도입되어 형성되었다. 이는 기존 YOLOv2에 비해 다소 느려졌지만 높은 정확도를 가져다준다. 아래의 그림 18과 같이 Residual Skip connection과 Up-sampling을 통해 다른 스케일로 이루어진 3가지의 검출기로 구성되어 있다. 이를 통해 이전 버전의 문제인 작은 물체 감지의 낮은 정확도를 해결하는 것에 도움을 주었다. Convolutional module은 Conv. 1x1과 Conv. 3x3 Kernel 5개로 구성되어 있다.



<그림 18> YOLOv3 구조

YOLOv3-tiny는 아래의 그림 18과 같이 기존 Darknet19의 Backbone인 Convolution layer와 Maxpool layer로 구성되어 있다. 또한 YOLOv3의 디텍더는 3개의 scale로 검출하는 것에 반해 YOLOv3-tiny의 검출할 디텍더는 2개의 scale로 형성되어 있다. 기존 YOLOv3는 53개의 Convolution layer를 포함한 Backbone과 함께 총 106개의 layer로 구성되어있다. YOLOv3-tiny는 7개의 Convolution layer를 포함한 총 23개의 Layer로 구성되어 있다. 따라서 효율적인 구조를 통해 성능의 정확도는 감소되었지만 검출 시간을 단축시키는 효과를 가져왔다.

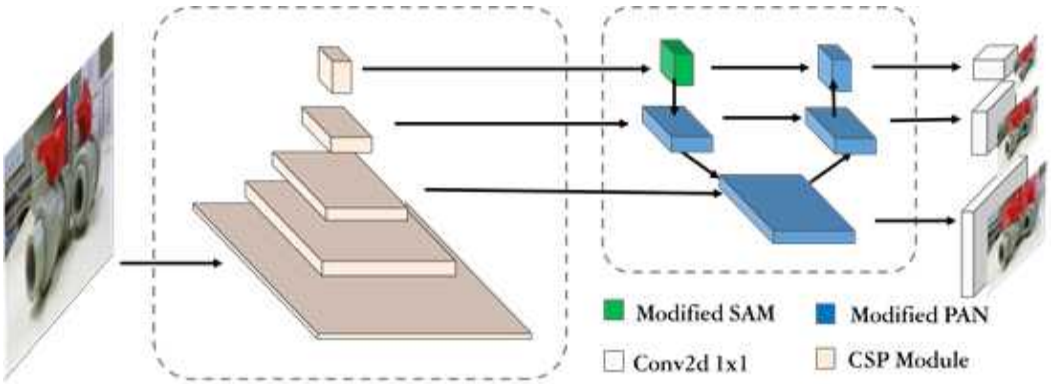


<그림 19> YOLOv3-tiny 구조

2) YOLOv4

YOLOv4 는 이전 버전인 YOLOv3 대비 혁신적인 구조적 변화는 없지만, 차이점으로 가장 두드러지는 것은 최신의 딥러닝 학습 구조 및 기법을 적용하여 성

능 향상을 도모하였다. Activation function 부분에서는 주로 Mish를 채택하였고, Regularization부분에서는 랜덤으로 Activations을 Dropout 시키지 않고, 상호연관성을 지닌 밀집된 구역을 Dropout 시키는 DropBlock을 적용하였다. Data augmentation 부분에서는 Mosaic, Self-Adversarial Training이 적용되었다. 아래의 그림 20와 같이 CSP module은 BiFPN에서 사용된Multi-Input Weighted Residual Connections(MIWRC)과 Cross Stage Partial connections (CSP)기반으로 Backbone을 재구성하여 CSPdarknet53을 구축하였다. CSPdarknet53과 더불어 SPP block과PAN block, SAM block 등을 추가하여 아키텍쳐를 구성하였다. 이 외에도, Cosine annealing scheduler, Optimal hyper-parameters, Random training shapes 와 같이 최신 적용시켰다. 이를 통해 YOLOv3에 비해 비슷한 속도를 유지하고 정확도를 높이는 성과를 달성하였다[19][20][21][22][23][24][25][26][27].



<그림 20> YOLOv4 구조

2. 딥러닝 기반 학습

딥러닝 학습을 위해 GPU 서버를 총 2대를 이용하여 진행하였다. 하나는 Tesla V100로 2개로 구성되어 있고, 나머지는 Tesla p100 2개, Titain xp 2개로 구성되어 있다. GPU의 성능에 따라 Batch, Mini-batch 사이즈를 달리하여 학습하였고 파라미터 설정 및 학습 방식은 아래의 설명과 같이 진행하였다.

1) Parameter setup

딥러닝 학습을 진행하기 위해, Hyper-parameter 설정은 다음과 같이 하였다: Class가 2개인 점을 고려하여 학습 Step은 12,000으로 하였고, Step decay learning rate는 초기에 0.01을 적용하고, Learning rate를 감소를 위한 학습 Step의 80,90% 단계에서 0.1 을 곱하였다. Momentum는 0.949, Weight decay 0.0005로 설정하였다. 64 Batch size와 32 또는 16의 Mini batch size를 정하였다.

2) Training method

YOLOv3 학습 시에 다음과 같은 방안을 적용되었다. YOLO9000에서 적용된 Dimension clusters를 활용하여 Box prediction를 진행하였다. 이는 K-means clustering을 실행하여 최적의 Anchor box의 종횡비와 크기를 찾도록 하였다. 이를 통해 IoU 정확도를 높이고자 하였다. 추가적으로 Batch normalization과 Multi-scale 학습 등이 적용되었다.

YOLOv4 학습동안 초기 학습 10% 동안 최적 Hyper-parameter선택을 실시하는 Genetic algorithm 이용하였고, 또한 특정 이미지 픽셀에 최적화된 Optimal anchor를 추가적으로 적용하여 학습하였다. Bounded box regression의 IoU loss를 향상시키기 위해 Ground true BBox와 Prediction BBox를 둘다 포함하는 가장 작은 BBox를 찾아내는 GIOU와 중심점의 거리를 고려한 DIOU를 포괄하는 CIOU를 사용하였다. 기존 YOLOv3에 적용된 Batch normalization보다 효과적인 Cross mini-Batch Normalization가 적용되었다. 그 외에도 Sigmoid activation을 위한 Class label smoothing, 4장의 이미지를 조합하여 한 번에 훈련하는 Mosaic data augmentation, 학습 중간에 더디게 학습될 수 있는 정체 구간에서 빠르게 벗어나게 해줄 Cosine Annealing Learning Rate Scheduling 등이 적용되었다 [28][29][30][31].

3. 정확도 평가

객체 인식 알고리즘의 성능 평가를 위해선 Precision과 Recall, IoU에 대해 이해가 필요하다. 물체를 검출할 때, 올바르게 검출이라는 기준은 IoU (Intersection over Union)으로 결정한다. 이는 기존의 실제 타겟의 위치인 픽셀 박스와 디텍터가 검출한 박스를 비교하는 것인데, 두 개 박스의 합집합 면적에 비해 두 개의 박스의 교집합 면적의 비율이 얼마냐에 따라 검출 유무를 결정이 된다.

실제값 (Ground Truth)	예측값 (Predict result)	
	Positive	Negative
Positive	TP (True Positive)	FN (False Negative)
Negative	FP (False Positive)	TN (True Negative)

<그림 21> 실제값과 예측값에 따른 정의

정밀도(Precision)는 디텍터가 검출한 결과 중 올바르게 검출한 비율을 말한다. 재현율(Recall)은 실제 값(Ground Truth) 중에 제대로 검출된 비율을 말한다. 예를 들어 실제 값이 진실인 것이 5개 중, 디텍터가 옳게 검출한 것이 4개라면 이는 Precision은 4/5로 0.8이다. 그러나 이 때, 디텍터가 총 10개를 진실이라고 검출하였다면 이는 Recall은 4/10로 0.4이다. 이는 실제 값에 비해 디텍터가 진실로 간주한 것이 상대적으로 많아 Precision은 높으나 recall은 낮다. 주로 Precision이 높으면 Recall이 낮아지는 경향이 있고, 반대 상황도 동일하다. 따라서 두 인자는 Trade-off 관계를 갖는다. 이와 같이 두 인자는 정확도와 안정성이 내재되는 중요한 인자로써 종합해서 성능을 평가하기 위해 AP(Average Precision)가 도입되었다.

$$\mathrm{Pr}ecision = \frac{TP}{TP+FP} \tag{5}$$

$$Recall = \frac{TP}{TP+FN} \tag{6}$$

1) AP

AP(Average Precision)는 PR 곡선에 의해 구해진 면적을 계산하여 하나의 수치로 표현한 것이다. PR곡선은 Confidence 레벨에 따라 Precision과 Recall 변화에 따라 그려지는 곡선이다. Confidence는 디텍터가 알고리즘에 의해 검출 결과에 대해 확신 정도를 의미한다. 따라서 Threshold 값에 비해 Confidence가 낮으면 검출 결과를 무시하는 것이다. 일반적으로 AP 평가 방안은 Pascal VOC, MS-COCO 와 같은 물체 인식의 평가방식으로 주로 활용하고 있다. 특히 각 검출하는 객체들의 각 AP를 평균한 것을 mAP라고 하며, 주로 위치 정확도를 포함한 IoU값에 따라 클래스 판별 정확도를 포함하는 mAP(@0.5), mAP(@0.75), mAP(@0.9)와 같이 성능을 평가한다.

2) Result

구축된 데이터셋은 학습용 데이터셋 및 테스트용 데이터셋으로 나누기위해, 무작위로 9:1 비율로 학습용과 테스트셋을 나뉘어 진행하여 딥러닝 학습 결과물인 모델을 검증하였다. mAP는 각 클래스의 AP값의 평균값을 의미한다. 이를 증식 데이터셋의 종류별, 딥러닝 객체 인식 종류별로 나뉘어 성능을 평가하였다. AP(@0.5)는 IoU값이 0.5이상 만족하는 정확도를 의미한다. AP(@0.75)는 IoU 0.75이상을 만족하는 정확도를 의미한다. 이는 사물의 클래스 정확도뿐 아니라 이미지 내의 위치 정확도 또한 반영된 정확도를 의미한다. YOLOv4 구조를 통해 학습을 실시하였을 때, 아래의 표 4와 같이 결과를 추출하였다. 왼쪽 Dataset

별로 AP의 정확도를 추출하였다. 기존에 온라인에서 수집된 데이터셋으로 학습하였을 때, AP는 63.21%의 정확도를 갖는다. 이에 비해 수집 이미지와 Color transform이 적용된 이미지를 통해 학습된 DCGAN, WGAN 각각의 생성 이미지로 이루어진 데이터셋으로 학습하였을 때, 기존 데이터셋 대비 3~6% 이상의 정확도가 향상되었다. 또한 SRGAN을 포함한 모든 GAN를 통해 생성한 데이터셋은 AP(@0.5) 기준 70%의 정확도를 달성하였다. 제안된 솔루션 기반의 데이터셋으로 학습되어진 모델을 비교하였을 때, 약 20% 이상의 높은 정확도를 얻었다. 그리고 AP(@0.75) 기준으로 기존 수집된 데이터셋으로 학습한 결과로 42.36% 정확도만 보이지만, 증식 데이터셋을 적용한 결과 최소 5%~30% 이상의 정확도를 향상시켰다.

<표 4> 데이터셋별 512x512 입력이미지 기반 YOLOv4 정확도

Dataset					Accuracy			
Origin	DCGAN	WGAN	SRGAN	Manipulations	AP(@0.5)	개당 정확도 증가율	AP(@0.75)	개당 정확도 증가율
√					63.21		42.36	
√	√				66.26 (+3.05)	+0.00287	47.74 (+5.38)	+0.0051
√		√			66.50 (+3.29)	+0.00247	48.59 (+6.23)	+0.0045
√	√	√	√		69.95 (+6.74)	+0.0014	50.88 (+8.52)	+0.0022
√	√	√	√	√	87.59 (+24.38)	+0.000186	76 (+33.64)	+0.000265

그리고 YOLOv3를 적용하였을 때, 정확도는 아래의 표 5와 같이 얻을 수 있었다. YOLOv4와 동일하게 데이터셋을 분류하여 학습을 실시하였다. 기존에 수집된 데이터셋 기반으로 약 61% mAP(@0.5)를 달성하였고 DCGAN기반의 생성된 데이터셋은 2%, WGAN기반의 데이터셋은 약 3%의 정확도 향상을 가져왔다. 또한 모든 GAN 기반의 데이터셋은 68% 정확도를 달성하였다. 제안된 솔루션 기반의 데이터셋으로 학습되어진 모델을 비교하였을 때, 약 20% 이상의 높은 정확도를 얻었다. mAP(@0.75)를 기준으로는 약 30%의 정확도를 달성하였기에 위치 정확도 또한 향상 효과를 보임을 알 수 있다. 이전의 YOLOv4와 비교하였을 땐 AP(@0.5) 기준으로 2~5%의 정확도가 저하되는 것을 확인하였다. 또한

AP(@0.75) 기준으로 보면 5~10%의 정확도가 낮아짐으로 위치정확도가 YOLOv3에 비해 YOLOv4의 성능이 향상됨을 알 수 있었다.

<표 5> 데이터셋별 512x512 입력이미지 기반 YOLOv3 정확도

Dataset					Accuracy			
Origin	DCGAN	WGAN	SRGAN	Manipulations	AP(@0.5)	개당 정확도 증가율	AP(@0.75)	개당 정확도 증가율
√					61.75		37.25	
√	√				63.83 (+2.08)	+0.00196	45.56 (+8.31)	+0.0076
√		√			64.18 (+2.43)	+0.00175	45.73 (+8.48)	+0.0061
√	√	√	√		68.06 (+6.31)	+0.0013	46.37 (+10.12)	+0.0021
√	√	√	√	√	85.31 (+23.56)	+0.000213	66.45 (+29.8)	+0.00027

그리고 YOLOv3-tiny를 적용하였을 때, 정확도는 아래의 표 6와 같이 얻을 수 있었다. YOLOv3과 YOLOv4와 동일하게 데이터셋을 분류하여 학습을 실시하였다. 수집된 데이터셋(Origin) 기반으로 약 50.28% mAP(@0.5)를 달성하였고 DCGAN기반의 생성된 데이터셋은 54.74%, WGAN기반의 데이터셋은 약 53.49%의 정확도 향상을 가져왔다. 또한 모든 GAN 기반의 데이터셋은 56.97% 정확도를 달성하였다. 제안된 솔루션 기반의 데이터셋으로 학습되어진 모델을 비교하였을 때, 약 67.96%의 높은 정확도를 얻었다. mAP(@0.75)를 기준으로는 약 15.59%의 정확도가 상당히 낮아짐을 확인하였다. 이전의 YOLOv3과 비교하였을 땐 AP(@0.5) 기준으로 18%의 정확도가 저하되는 것을 확인하였다. 또한 AP(@0.75) 기준으로 보면 50%의 정확도가 대폭 낮아짐을 확인하였다. 상대적으로 YOLOv4, YOLOv3와 다르게 YOLOv3-tiny는 위치 정확도가 상당히 부정확함을 알 수 있었다. 이는 구조를 단순히 하여 속도 측면을 집중적으로 고려하였기에 위와 같은 성능 저하가 발생한 것을 알 수 있다.

<표 6> 데이터셋별 512x512 입력이미지 기반 YOLOv3-tiny 정확도

Dataset					Accuracy			
Origin	DCGAN	WGAN	SRGAN	Manipulations	AP(@0.5)	개당 정확도 증가율	AP(@0.75)	개당 정확도 증가율
√					50.28		12.91	
√	√				54.74 (+4.66)	+0.00438	13.95 (+1.04)	+0.00098
√		√			53.49 (+3.21)	+0.00231	14.08 (+1.17)	+0.00084
√	√	√	√		56.97 (+6.69)	+0.0014	14.19 (+1.28)	+0.00026
√	√	√	√	√	67.96 (+17.68)	+0.00016	15.59 (+2.68)	+0.000024

위의 결과와 같이 본 논문에서 제안하는 증식 데이터셋은 세 가지의 모든 딥러닝 구조로 학습되어 성능 검증을 실시하였다. 세 개의 딥러닝 구조로 학습된 모델은 상대적으로 GAN 생성된 이미지를 추가하였을 때에 비해 Manipulation을 추가 하였을 때 인식 정확도의 증가폭이 더 큼을 알 수 있었다. 이는 위와 같은 성능 향상의 근거로는 기존의 수집된 데이터셋은 총 9642개이고, GAN 알고리즘으로 생성된 이미지는 4906개, Manipulation 생성 이미지는 93568개, 최종 증식된 데이터셋은 117515개로 획기적으로 증축되었기 때문이다. 따라서 Manipulation의 생성 이미지가 상대적으로 많기 때문에 증가폭이 더 컸다. 그러나 YOLOv4 기준 GAN 추가된 이미지 1개당 증가 비율은 0.0013%, Manipulation 추가된 이미지 1개당 증가 비율은 0.00018%로 GAN이 10배 이상으로 증가함을 알 수 있었다. 기존 수집된 이미지에서 벗어나 딥러닝 증식 기법을 통해 새롭게 생성된 증식 이미지를 학습하였기에 편향된 학습 모델을 방지하였음을 의미한다.

또한 특정한 물체 인식을 위한 데이터셋 구축하는데 상당한 시간이 소모되는 점을 고려하였을 때, 데이터셋 구축의 소요 시간에 상당한 효율성이 있음을 알 수 있다. 구체적으로 온라인에서 이미지 수집은 웹 크롤링, 영상에서 이미지 추출과 색출하는데 소요시간은 약 반나절이 걸렸다. 이미지 내의 물체 정보를 표기하는 라벨링 작업은 단 하루가 소요되었다. DCGAN, WGAN, SRGAN, 객체 인식 딥러닝 구조에 수집 이미지를 학습하여 최적 모델을 생성하였을 때, 하나의 물체에 딥러닝 하나의 구조 당 3~7일이 소요된다. 이는 GPU 서버 성능에 따라 학습 기간이 상이하다. 이와 같이 학습된 모든 GAN 학습 모델로 새로운 이미지를 생성하는데 1일이 넘지 않는다. 또한 이미지 회전, 블러 등과 같은 이미지 Manipulation이 적용된 이미지를 생성하기까지 프로그래밍 시간을 제외하고

3시간 안에 달성할 수 있었다. 기존에 제공되는 공유 데이터셋이 구축되기에 수년의 노력이 필요한 점을 들어, 본 논문에서 수행했던 데이터셋 구축 과정 시간은 최대 한 달의 소요 시간과 한 명의 인력을 통해 데이터셋 구축이 가능하기에 실제 환경에서 접목하기에 유용함을 알 수 있었다.

IV. 로봇 기반의 실증 테스트

실제 환경에서 물체 인식 시스템 평가 및 검증을 위해 로봇의 하드웨어와 소프트웨어 구성을 소개하였다. 특히 로봇의 딥러닝 알고리즘의 실시간 연산을 위한 임베디드 시스템 및 카메라를 중점으로 언급하도록 한다. 로봇에게 있어서 카메라 기반의 영상 처리는 사람의 눈과 같이 주변 환경의 인지 및 사물의 카테고리를 판별하는 것에 핵심적인 역할을 한다. 이를 위험 요소를 지니는 개방형 환경에서의 대처하기 위해 딥러닝 기술을 적용하기 위해서는 방대한 연산을 처리할 수 있는 GPU 기반의 임베디드 시스템이 탑재되는 것이 필수적이다. 또한 다양한 환경에 적응 가능한 안정적인 로봇 플랫폼을 기반으로 아래의 그림 22과 같이 구성하였다. 다음 표 7은 UGV 및 카메라 관련 사양을 표시하였다. 그리고 다음의 표 8은 각기 다른 사양을 지닌 임베디드 시스템 기반의 연결 플랫폼 구축을 나열하였다.

1. Robot Configuration

1) Robot hardware : AECOBOT

우리는 다양한 환경에서의 견고하게 작업이 가능한 Clearpath 사의 Husky 4륜 구동 이동 로봇을 기반으로 시스템을 설계하였다. 이를 기반으로 Intel 사의 RGB-D 카메라와 임베디드 보드를 부착하여 딥러닝 기반의 영상 처리를 활용하여 객체 인식 구축하였고, 그 외에도 3D CAD 등을 이용하여 라이다, GPS, LattePanda, 유무선 공유기, 배터리를 부착하여 다양한 기능을 구현이 가능하도록 구축하였다. 이를 활용하여 실제 환경에서 테스트를 진행하도록 한다[32].

<그림 22> 로봇 하드웨어 구성

목표물 인식을 위한 영상 데이터 취득은 Intel 사의 d435i 카메라를 사용하였다. 1920x1080 화소의 30fps 데이터를 송신하고, 화각(Field of View) 중 수평각도는 69.4 Deg, 수직각도는 42.5 Deg, 대각선각도는 77 Deg 정도의 성능을 지니고 있다. 따라서 Embedded 시스템과 USB 3.0 포트를 활용하여 연결하여 데이터를 취득하였다[33].

<표 7> 로봇 시스템 구성표

	module	**detail**
UGV	Battery	Li-ion 25.2V-68Ah
	Embedded	Nano, TX2, Xavier
	Internet Modem	ASUS
Camera	Camera	Intel d435i
	RGB Resolution	1920x1080
	Frame rate	30fps
	FOV(H x V x D)	69.4x42.5x77
	Connectors	USB 3.0 Type-C

2) Embedded System

이동 로봇의 딥러닝 기반의 객체인식을 구현하기 위해 방대한 연산 처리를 빠른 시간 내에 처리가 가능 GPU가 포함된 NVIDIA JETSON 의 Nano, TX2, Xavier를 선택하였다. 이는 저전력 운영으로 딥러닝 및 영상 정보의 연산 처리의 높은 성능을 제공합니다. 딥러닝 연산을 위한 TensorRT, OpenCV, CUDA toolkit, CUDNN 등을 지원합니다. NVIDIA 사에서 제공하는 대표적인 임베디드 보드 3가지 Nano, TX2, Xavier를 대상으로 하였으며, 이는 제한적인 코스트 부분에 따른 사양 차별화를 두어 효율적인 성능을 제공합니다[34][35][36].

먼저 Nano는 NVIDIA 사에서 제공하는 가장 저가라인의 제품으로, 오직 5W의 전력 소모로 딥러닝 알고리즘의 여러 신경망을 병렬로 실행하여 효율적인 성능을 제공한다. 이를 통해 이미지 분류, 물체 감지 및 음성 처리 등과 같은 어플리케이션이 적용가능 하도록 제작되었다. NVIDIA 사에서 고안한 GPU Architecture 인 Maxwell 구조를 채택하였고 128 코어로 이루어져 있다. 그 외에 4core의 ARM cpu 및 4 GB 메모리로 구성되어 있다. 이는 소형 로봇 및 제한적인 비용을 감안하였을 때, Nano를 사용하는 것은 현명한 선택중 하나가 될 수 있다.

TX2는 NVIDIA 사에서 AI 슈퍼 컴퓨터 용으로 제공하는 제품으로, 다양한 임베디드 보드의 기능을 제공하고 딥러닝 용 라이브러리 및 다양한 다중 프로세서 기능을 포함하는 SDK를 제공합니다. GPU architecture 인 Maxwell 구조를 채택하였고 128 코어로 이루어져 있다. 그 외에 4 Core의 Arm CPU 및 4 GB 메모리로 구성되어 있다. Maxwell 다음으로 발표한 Pascal 구조로 이루어져 있으며, Nano 대비 많은 CPU core 와 높은 전력을 이용하여 이미지 분류 및 검출의 성능 측면으로 볼 때 Nano의 약 2배의 처리 능력을 가지고 있다.

Xavier는 최근에 발표한 Volta architecture를 채택한 고성능의 제품으로, CPU 또한 앞의 임베디드 보드에 비해 많은 8개의 코어로 이루어져 있으며 Tensor core를 지원하는 임베디드 보드이다. 앞서 Nano의 이미지 분류 및 검출의 성능은 약 20배 이상의 능력을 가지고, tx2와는 약 10 배 이상의 성능을 제공하는 것으로 알려져 있다.

세 가지의 임베디드 시스템을 로봇(AECOBOT)에 탑재하여 작동할 수 있도록 Nano는 5V의 전원을 인가하였고, TX2, Xavier는 12V의 전원을 주었다. 이를 실제 환경에서 실시간 카메라의 영상 데이터를 활용하여 딥러닝 모델별로 인지 속도 성능을 비교하였다.

<표 8> 임베디드 보드 성능 비교

	Nano	**TX2**	**Xavier**
Architecture	128-core Maxwell @ 921Mhz	256-core Parscal @ 1.3Ghz	512-core Volta @ 1.37Ghz
CPU	4-core ARM A57 @ 1.43Ghz	4-core ARM A57 @ 2Ghz, 2- core Denver2 @ 2Ghz	8-core ARM Carmel v.8.2 @ 2.26 Ghz
Memory	4 GB LPDDR4, 25.6 GB/s	8 GB 128-bit LPDDR4, 58.3 GB/s	16GB 256-bit LPDDR4,137 GB/s
Tensor cores	-	-	64
Size (mm)	100 x 80 x 29	170 x 170 x 30	105 x 105 x 65
Power (W)	5 / 10	7.5 /15	10 / 15/ 30

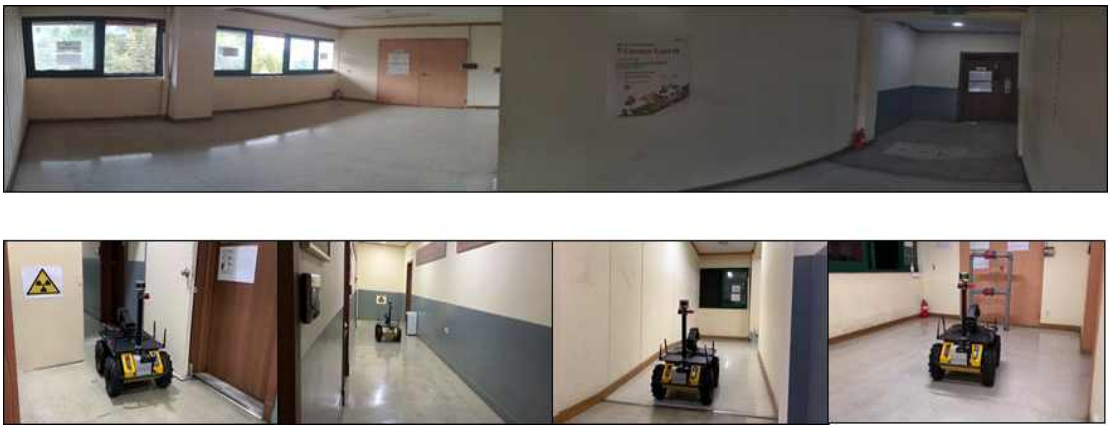
<그림 23> 임베디드 시스템 외형 (왼쪽 : Nano , 중간: TX2 , 오른쪽 : Xavier)

3) ROS & Communication

위의 그림 22과 같이 구성된 4 WHEELS 로봇을 기반으로 모든 소프트웨어 운영 및 데이터 관리를 ROS(Robot Operating System) 플랫폼을 사용하였다. ROS 운영체제를 통해서 이동 로봇의 카메라 데이터와 같이 센서 계측, 물체 인식의 결과인 사물 정보(종류, 개수)와 같은 데이터를 통합할 수 있는 관리 시스템을 구축하였고, 원격 모니터링 및 제어가 가능하도록 ASUS 사의 유무선 공유기를 통하여 WI-FI 6 기반의 최대 10Gbps의 속도를 지원하는 광대역을 구축하여 데이터 끊김 없이 원격 데스크탑에서 실시간으로 확인이 가능하도록 하였다. ROS 플랫폼을 이용하여 실제 환경에서 로봇을 주행하였고 Object detection 성능을 검증하였다.

2. Experiment

딥러닝 학습을 통해 생성되어진 모델을 실제환경에서 시스템 성능 검증을 하기 위해, 로봇(AECOBOT)에 위에서 소개한 다양한 임베디드 시스템을 탑재하여 테스트를 실시하였다. 실험 장소는 울산대학교 7호관 6층에서 진행하였다. 아래의 사진과 같이 방사능 표시판 2개, 밸브 1개를 이동체의 이동경로에 임의로 배치하여 실험을 실시하였다.

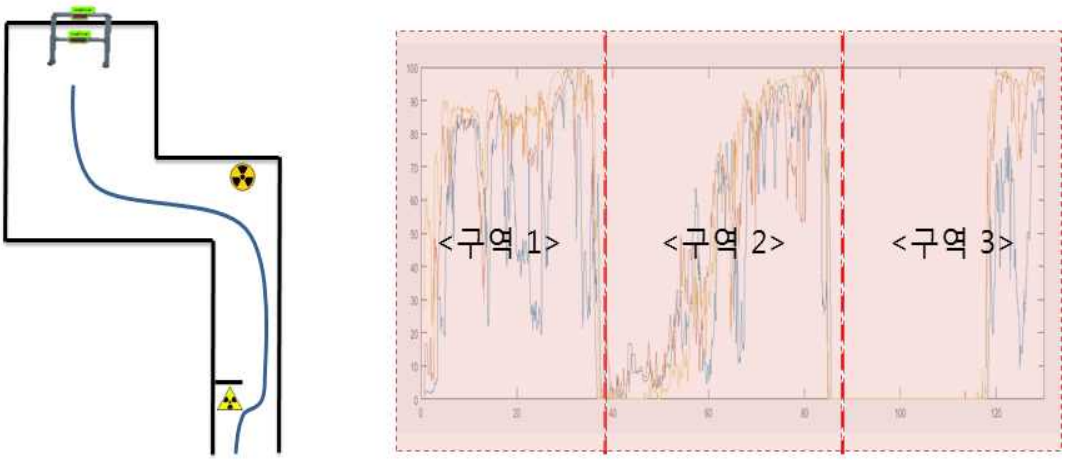


<그림 24> 7호관 6층 실내 환경

1) 목표물 인식 정확도 성능 검증

증식 데이터셋 기반의 인식 모델의 성능을 실제 환경에서의 평가를 위해, 각각의 데이터셋을 통해 생성된 모델별로 동일한 조건 속에서 테스트를 진행하였다. 아래의 그림 25와 같이 이동체가 주행을 실시하였고, 주행 시에 각각의 인식 대상물에 대한 그래프를 아래와 같이 추출하였다. 그래프의 y축 정확도(Accuracy)는 실제 환경에서의 추론(Inference) 시 YOLO 객체인식의 결과물인 Class confidence score를 의미한다. Class confidence score는 $P_r(Object)$과 $P_r(Class|Object)$ 곱으로 계산된다. $P_r(Object)$는 물체 위치 후보군(Bounding box) 추출 과정에서 Bounding box의 Confidence score를 예측값(물체 존재 확률)이고, $P_r(Class|Object)$는 물체가 각각의 Grid cell에서 물체가 Class 분류 예측값인 조건부 확률을 의미한다. 학습 시에는 해당 Score를 실제 Bounding box 위치와 비교 학습을 통해 모델 개선이 이루어진다. 따라서 테스트 시에 결정되는 Score는 모델이 인식한 결과인 Class 및 위치에 대한 확신을 수치로 표현한 값이다. 주로 50%의 Threshold 값을 기준으로 인식 유무를 결정한다.

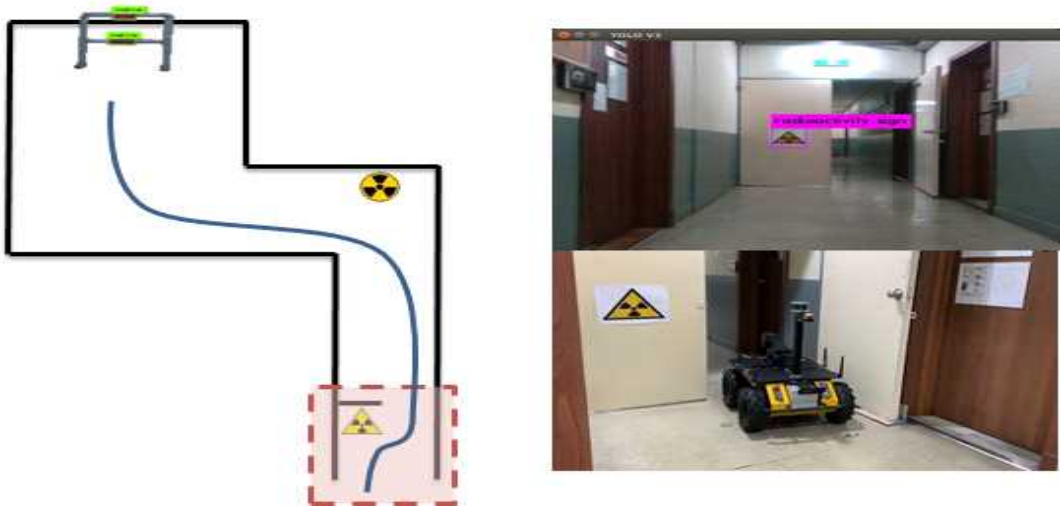
$$Class\ confidence\ score = P_r(Object) * P_r(Class|Object) \tag{7}$$



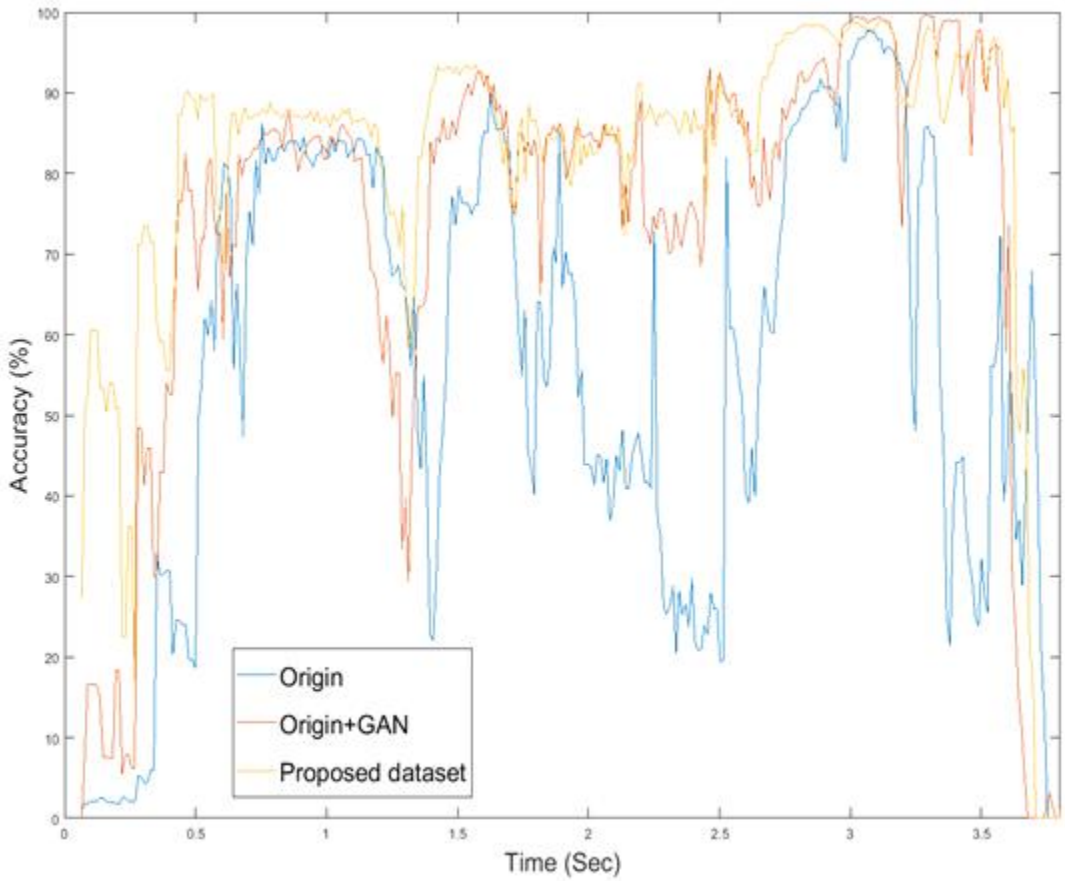
<그림 25> 실제 환경에서의 로봇 기반 정확도 추출

인식 대상물이 위치한 각 세 곳의 영역을 분리하여 정확도를 비교하였다. 파란색 선의 정확도는 수집된 데이터셋으로 학습된 인식 모델의 정확도이고, 빨간색 선은 수집된 이미지와 GAN을 통해 생성된 이미지를 포함한 데이터셋이다. 노란색 선은 수집된 이미지, GAN 생성 이미지, Manipulation 생성 이미지를 포함한 데이터셋으로 학습된 모델의 정확도이다. 구역 1에서 방사능 표시판1 에 대한 정확도로, 파란색의 정확도가 편차가 심하고 정확도 또한 떨어지는 그래프를 확인하였다. 그에 비해 빨간색의 정확도는 편차가 줄어들었고 상대적으로 높은 정확도를 보여주지만, 노란색 선에 비해 성능이 떨어짐을 알 수 있었다. 노란색 선

은 이동체의 직선 주행에 따라 환경 조건과 카메라의 위치가 변함에도 안정적인 정확도 결과를 보여주었다.



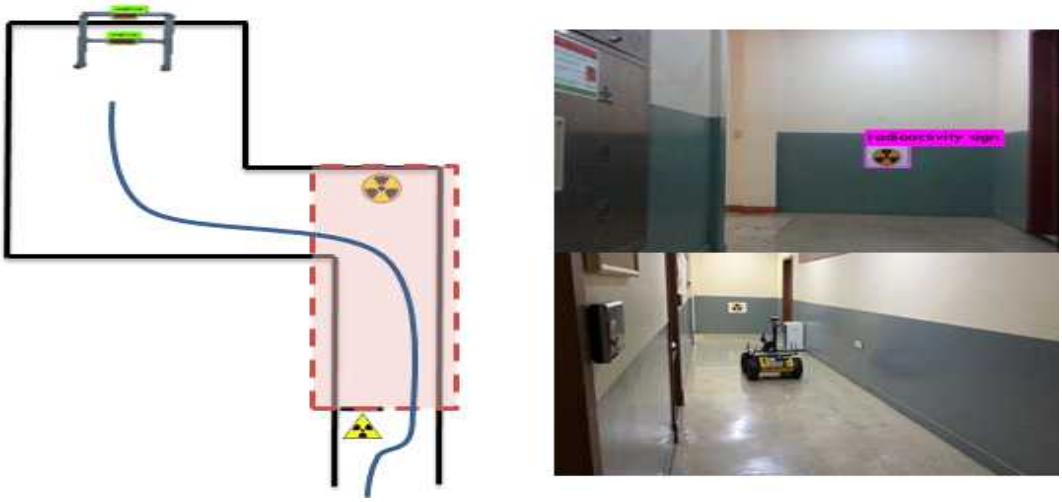
<그림 26> 구역 1에서의 방사능 표시판1 인식 및 주행 모습



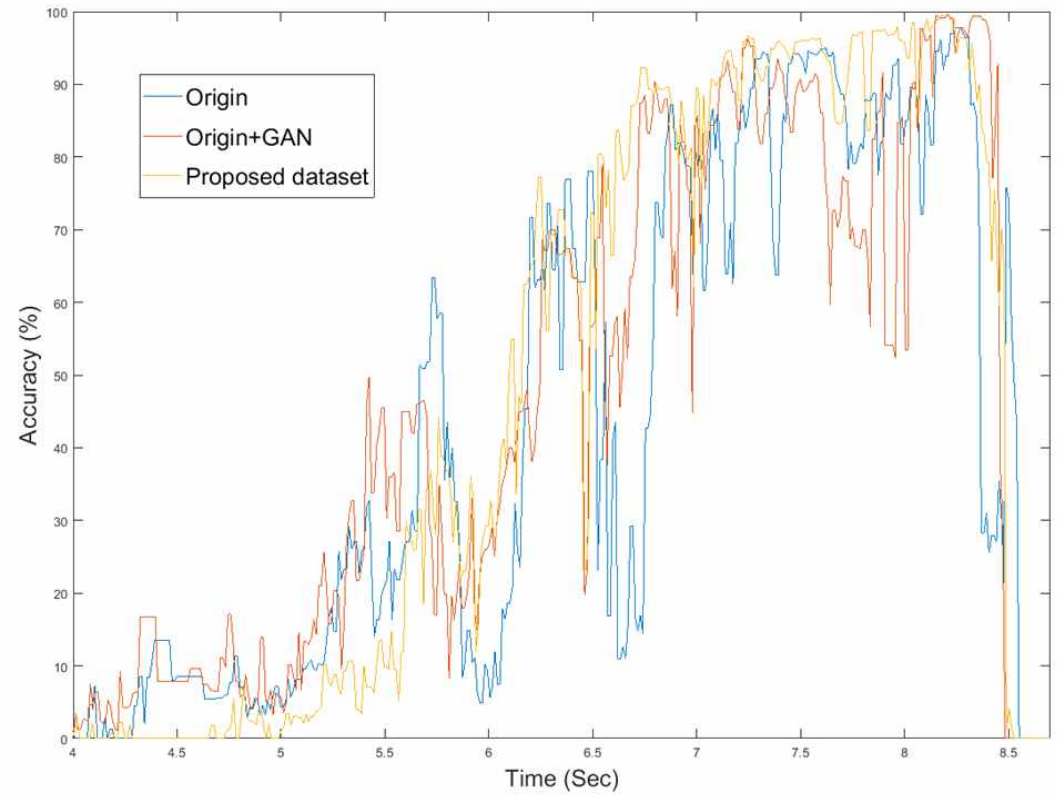
<그림 27> 데이터셋별 방사능표시판1 인식 정확도

그 후 구역2에 진입하여 구역1에 비해 방사능 표시판2에 대한 인식 정화도는

아래의 그래프와 같다. 약 15m 범위 밖에서는 세 개의 데이터셋에 학습된 모델은 모두 50% 이내의 정확도로 방사능 표시판을 검출하지 못하였다. 하지만 표시판에 근접하면서 정확도가 향상되었다. 로봇의 직선 주행임에도 빛과 같은 환경 변하에 따라 파란색의 정확도는 상당히 편차가 심함을 알 수 있었다. 빨간색의 정확도는 그에 비해 편차가 줄어들었음을 알 수 있었으나 노란색의 정확도에 비해 불안정하고 낮은 정확도를 보여주었다.

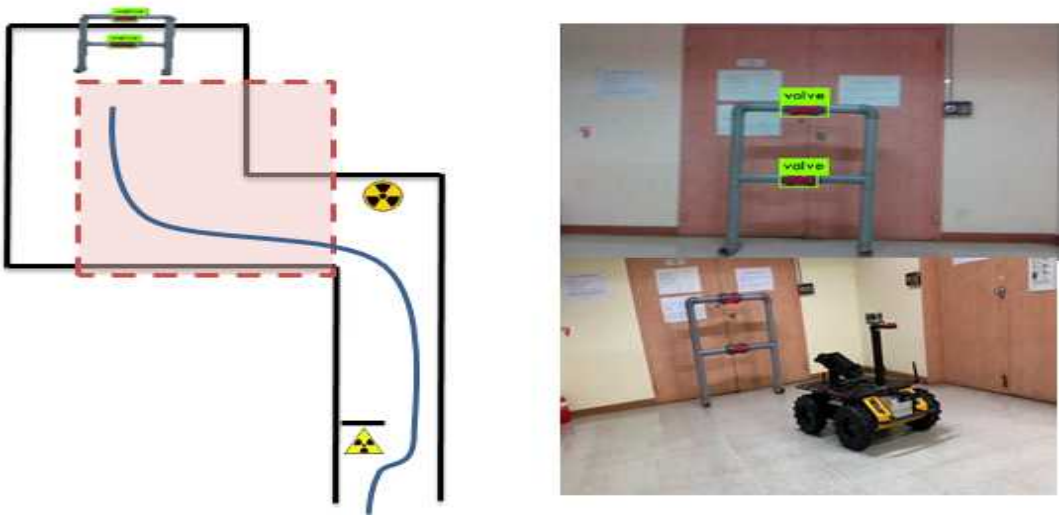


<그림 28> 구역 2에서의 방사능 표시판2 인식 및 주행 모습

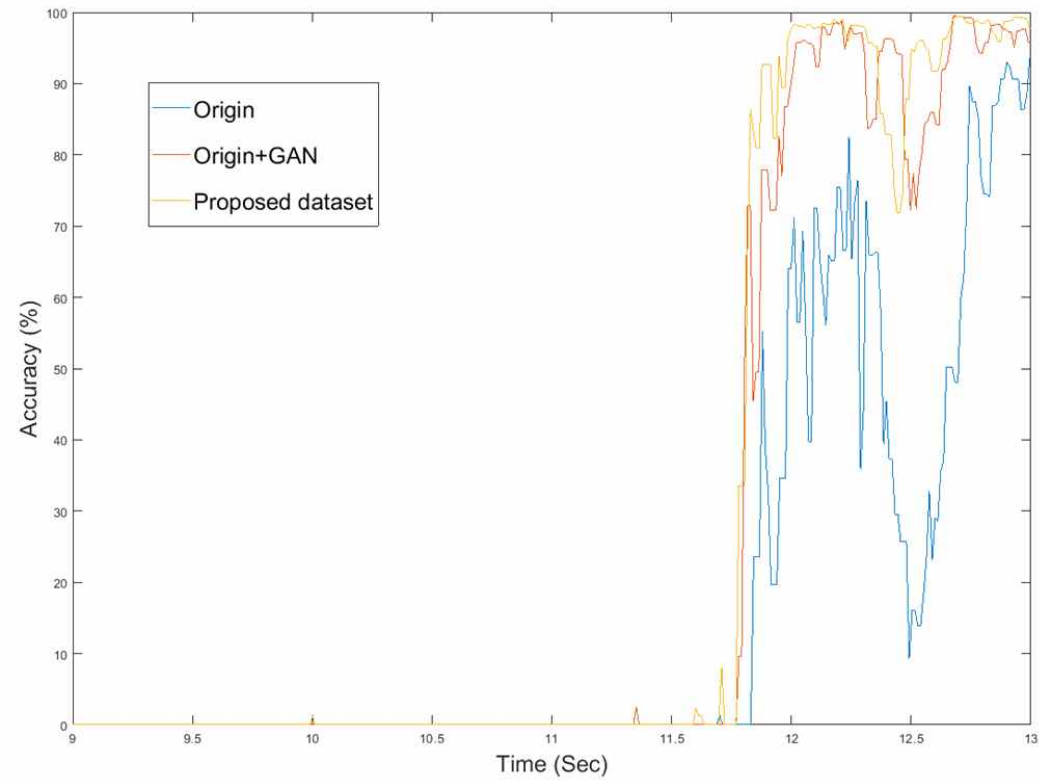


<그림 29> 데이터셋별 방사능표시판2 인식 정확도

구역 3에 진입하여 로봇이 밸브에 접근하는 상황으로, 종 방향 및 횡 방향 주행을 통해 이미지 내에 밸브가 들어오면서 정확도 변화 그래프이다. 수집된 데이터셋으로만 구성된 파란색의 정확도는 이동체의 횡 방향 주행에 따라 밸브의 정확도가 순간적으로 급격히 떨어짐에 비해, GAN이 포함된 데이터셋들은 약간의 정확도가 낮아지나 변화가 적었다. 수집된 이미지와 GAN 생성 이미지, Manipulation 생성 이미지로 구성된 데이터셋의 정확도인 노란색 정확도는 빨간색 정확도에 비해 높은 정확도를 유지하였다.



<그림 30> 구역 3에서의 밸브 인식 및 주행 모습



<그림 31> 데이터셋별 밸브 인식 정확도

알고리즘 깊이, 데이터셋의 양에 따라 정확도 차이를 실제 환경에서 테스트를 실시하였다. 이를 통해 알고리즘이 깊을수록, 데이터셋 양이 많을수록 인지 정확도 높아짐을 확인할 수 있었고, 이를 기반으로 상대적으로 로봇 주행에 따른 환경 변화와 카메라 위치 이동에 따른 인지 정확도가 높아져 가시거리를 넓히는 효과를 제공해 주는 것을 확인하였다.

2) 인지 속도 검증

카메라 기반의 목표물 인지 속도는 로봇의 안정적인 주행에 상당한 영향을 끼친다. 따라서 각 임베디드 시스템별로 실시간 영상 이미지가 입력으로 들어올 때마다 딥러닝 연산 수행하는 것을 평가를 위해 물체 인식 FPS(Frame Per Secound)를 검증하였다. 각 소프트웨어 및 임베디드 시스템의 설정에 따라 성능의 차이는 존재한다. 실험 시 임베디드의 GPU 최대 성능을 가지도록 설정하였고, 심층 신경망 연산 가속을 위한 CUDNN 라이브러리 등을 사용하였다. 이 외에도 임베디드 시스템의 주입되는 Power supply 크기에 따라 성능의 변화가 존재함을 확인하였다. CNN기반의 목표물 인식의 Input pixel resolution이 연산량과 관련이 있기에 성능에 지대한 영향을 끼친다. 따라서 인식 구조별 실험과 인식 신경망의 Input size를 달리하여 진행하였다.

<표 9> YOLOv3 기준 임베디드 시스템별 인지 속도

YOLOv3	**Input network resolution**	**FPS**
Nano	416x416	2-3
TX2	416x416	7-8
Xavier	**416x416**	**13-15**

<표 10> YOLOv3-tiny 기준 임베디드 시스템별 인지 속도

YOLOv3-tiny	**Input network resolution**	**FPS**
Nano	416x416	12-13
TX2	416x416	23-25
Xavier	**416x416**	**50-54**

위의 표 9와 같이 입력 이미지 픽셀 크기 416x416으로 주입하였을 때, YOLOv3 기준으로 Nano는 약 2-3 FPS, TX2는 7-8 FPS, Xavier는 13-15 FPS를 확인하였다. 또한 YOLOv3-tiny 기준으로 Nano는 약 12-13 FPS, TX2는 23-25 FPS, Xavier는 50-54 FPS를 확인하였다. 이와 같이 실시간 인식이 가능하도록 안정적인 성능을 보장하기 위해 Xavier 임베디드 시스템 기반으로 객체 인식 구현에 적합하고, 조건에 따라 TX2, Nano를 채택하는 것이 효율적이다.

<그림 32> 실제 환경에서 실시간 검출 이미지

V. 결론 및 향후 계획

본 논문에서 우리는 커스텀 데이터셋의 문제점을 보완하기 위한 증식 데이터셋 구축을 위한 증식 솔루션을 제안한다. 구축되어진 데이터셋 기반으로 학습하여 모델의 성능 평가 및 실제 환경에서 검증까지 실시하였다. 특히 커스텀 데이터셋 수집부터 물체의 정보를 포함한 Annotation 파일을 생성하는 라벨링 작업을 실시하였고, 이를 기반으로 회전, 블러, 랜덤 노이즈 등의 이미지 Manipulation과 딥러닝 기반의 DCGAN, WGAN, SRGAN과 같은 GAN 알고리즘들로 구성된 증식 솔루션을 제안하여 데이터셋을 구축하였다. 데이터셋, 객체 인식 알고리즘 별로 딥러닝 학습을 실시하였고, 기존 수집 데이터셋 대비 증식 데이터셋 정확한 인지 성능을 mAP 기준으로 15-20% 향상시켰다. 또한 증식 기법에 따라 증식 이미지 개수 당 정확도 향상 비율에 대해 분석하였다. 기존 연구의 Manipulation 기법에 비해 GAN 증식 기법의 향상 비율이 10배 이상 증가함을 확인하였다. 기존 연구 방안에 비해 GAN 기법을 적용하면서 수집된 데이터셋에 편향되지 않고 성능 향상에 도움이 되었음을 알 수 있었다. 또한 다양한 이동체의 임베디드 시스템을 활용하여 실제 환경에서의 테스트를 진행하고 인식 정확도와 속도 평가를 실시하였다. 실내 환경 테스트를 통해 기존 수집 데이터셋 기반의 학습된 모델에 비해 증식 데이터셋의 인식 정확도와 안정성 또한 향상되었음을 확인하였다. 또한 다양한 환경에서 임베디드 시스템별, 딥러닝 모델별로 실시간 인식 속도를 확인할 수 있었다. 앞으로 이 결과를 토대로 이미지 증식 방안의 효율적인 프레임 워크 방안과 융합 알고리즘을 통한 증식 시스템 단순화 방안에 대해 연구할 것이다. 이 외에도 제안하는 시스템에 적합한 딥러닝 알고리즘 구조 제안 및 극한의 환경에서 성능 검증을 실시할 것이다.

참고문헌

[1] Author 1, Zhaowei Cai; Author 2, Nuno Vasconcelos. Cascade R-CNN: Delving into High Quality Object Detection. 2017, arXiv:1712.00726.

[2] Author 1, Joseph Redmon; Author 2, Santosh Divvala; Author 3, Ross Girshick; Author 4, Ali FarhadiTitle of the chapter. You Only Look Once: Unified, Real-Time Object Detection. 2015, arXiv :1506.02640.

[3] Author 1, Joseph Redmon; Author 2, Ali Farhadi. YOLO9000: Better, Faster, Stronger. 2016, arXiv:1612.08242.

[4] Author 1, Joseph Redmon; Author 2, Ali Farhadi. YOLOv3: An Incremental Improvement. 2018, arXiv:1804.02767.

[5] Author 1, Alexey Bochkovskiy; Author 2, Chien-Yao Wang;Author3,, Hong-Yuan Mark Liao; YOLOv4: Optimal Speed and Accuracy of Object Detection. 2020, arXiv:2004:10934.

[6] Author 1, Sangdoo Yun; Author 2, Dongyoon Han; Author 3, Seong Joon Oh; Author 4, Sanghyuk Chun. CutMix: Regularization Stratgy to Train Strong Classifiers with Localizable Features. 2019, ICCV

[7] Author 1, Ian J.Goodfellow; Author 2, Jean Pouget-Abadie; Author 3, Mehdi Mirza; Author 4, Bing Xu, David

[8] Author 1, Ross Girshick; Author 2, Jeff Donahue; Author 3, Trevor Darrell; Author 4, Jitendra Malik. 5, Rich feature hierarchies for accurate object detection and semantic segmentation. 2014, CVPR.

[9] Author 1, Ross Girshick; Fast R-CNN. 2015 arXiv:1504.08083.

[10] Author 1, Shaoqing Ren; Author 2, Kaiming He; Author 3, Ross Girshick; Author 4, Jian Sun. Faster R-CNN: TOwards Real-Time Object Detection with Region Proposal Networks. 2015, arXiv:1506:01497.

[11] Author 1, Alec Radford; Author 2, Luke Metz; Author 3, Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. 2015, CoRR, abs/1511.064

[12] Author 1, Martin Arjovsky; Author 2, Soumith Chintala; Author 3, Leon Bottou. Wasserstein GAN. 2017, arXiv:1701.07875.

[13] Author 1, Christian Ledig; Author 2, Lucas Theis; Author 3, Ferenc Huszar; Author 4, Jose Caballero; Author 5, Andrew Cunningham. Photo-Realistic Single Image Super-Resolution Using a Generative

Adversarial Network. 2016, arXiv:1609.04802.

[14] Author 1, Wei Liu; Author 2, Dragomir Anguelov; Author 3, Christian Szegedy; Author 4. , Scott Reed; Author 5, , Cheng-Yang Fu; Author 6. , Alexander C. Berg; SSD: Single Shot MultiBox Detection. 2016, Arxiv:1512.02325.

[15] Author 1, Tsung-Yi Lin; Author 2, Priya Goyal; Author 3, Ross Girshick; Author 4, Kaiming He;Author5, Piotr Dollár. Focal Loss for Dense Object Detection. 2017, arXiv:1708.02002.

[16] Author 1, Mingxing Tan; Author 2, Quoc V. Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In Proceedings of International Conference on Machine Learning (ICML), 2019.

[17] Author 1, Kaiming He; Author 2, Georgia Gkioxari; Author 3, Piotr Dollar; Author 4, Ross Girshick. Mask R-CNN. 2018 arXiv:1703.06870.

[18] Author 1, Ze Yang; Author 2, Shaohui Liu;Author3, Han Hu; Author 4, Liwei Wang;Author5, Stephen Lin; RepPoints: Point Set Representation for Object Detection. 2019 arXiv:1904.11490.

[19] Author 1, Diganta Misra. Mish: A self regularized nonmonotonic neural activation function. arXiv preprint arXiv:1908.08681, 2019.

[20] Author 1, Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. DropBlock: A regularization method for convolutional networks. In Advances in Neural Information Processing Systems (NIPS), pages 10727 - 10737, 2018.

[21] Author 1, Chia-Jung Chou; Author 2, Jui-Ting Chien; Author 3, Hwann-Tzong Chen. Self Adversarial Training for Human Pose Estimation. arXiv preprint arXiv:1707.02439, 2017

[22] Author 1, Mingxing Tan; Author 2, Ruoming Pang; Author 3, Quoc V Le. EfficientDet: Scalable and efficient object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[23] Author 1, Chien-Yao Wang; Author 2, Hong-Yuan Mark Liao; Author 3, Yueh-Hua Wu; Author 4, Ping-Yang Chen; Author 5, Jun-Wei Hsieh; Author 6, I-Hau Yeh. CSPNet: A new backbone that can enhance learning capability of cnn. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPR Workshop), 2020.

[24] Author 1, Kaiming He; Author 2, Xiangyu Zhang; Author 3, Shaoqing

Ren; Author 4, Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 37(9):1904 – 1916, 2015.

[25] Author 1, Shu Liu; Author 2, Lu Qi; Author 3, Haifang Qin; Author 4, Jianping Shi; Author 5, Jiaya Jia. Path aggregation network for instance segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 8759 – 8768, 2018.

[26] Author 1, Sanghyun Woo; Author 2, Jongchan Park; Author 3, Joon-Young Lee; Author 4, In So Kweon. CBAM: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), pages 3 – 19, 2018.

[27] Author 1, Ilya Loshchilov; Author 2, Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983, 2016.

[28] Author 1, Hamid Rezatofighi; Author 2, Nathan Tsoi; Author 3, JungYoung Hwak. Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression. 2019, arXiv:1902.09630.

[29] Autohr 1, Zhaohui Zheng; Author 2, Ping Wang; Author 3, Wei Liu; Author 4, Jinze Li; Author 5, Rongguang Ye; Author 6, Dongwei Ren. Distance-IoU Loss: Faster and better learning for bounding box regression. In Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), 2020.

[30] Author 1, Zhaohui Zheng; Author 2, Ping Wang; Author 3, Wei Liu; Author 4, Jinze Li; Author 5, Rongguang Ye; Authot 6, Dongwei Ren. Distance-IoU Loss: Faster and better learning for bounding box regression. In Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), 2020.

[31] Author 1, Zhuliang Yao; Author 2, Yue Cao; Author 3, Shuxin Zheng; Author 4, Gao Huang; Authot 5, Stephen Lin. Cross-iteration batch normalization. arXiv preprint arXiv:2002.05712, 2020.

[32] Clearpathrobotics husky. Available online: https://clearpathrobotics.com/husky-unmanned-ground-vehicle-robot/

[33] Intel d435i. Available online: https://www.intelrealsense.com/depth-camera-d435i/

[34] Jetson Nano. Available online: https://developer.nvidia.com/embedded/jetson-nano-developer-kit/ (accessed

on 3 March 2019).

[35] Jetson TX2. Available online: www.nvidia.com/en-gb/autonomous-machines/embedded-systems/jetson-tx2/ (accessed on 3 March 2019).

[36] NVidia JETSON AGX XAVIER: The AI Platform for Autonomous Machines. Available online: www.nvidia.com/en-us/autonomous-machines/jetson-agx-xavier/ (accessed on 3 March 2019).

ABSTRACT

Design and Implementation Object dectection framework based Deep learning with Robot system

Lee Dong Hee

Graduate school of Electrical Engineering
University of Ulsan
Ulsan, Korea

Recently, the development of artificial intelligence has led to rapid environmental changes in various industries. Already, many factory lines are being replaced by automated equipment and unmanned robots on behalf of humans, and demand for them is growing. In addition, beyond limited environments such as self-driving vehicles and unmanned delivery robots, the boundaries of our daily radius are also breaking down, accounting for an increasing portion. Therefore, technical demands that can be responded to in such diverse environments should be steadily raised and robot movements in unexpected environments should be robust. As a key technology, it is essential to build a robot platform that can perform various tasks and image object recognition technology based on deep learning.

Research such as improving deep learning structures and optimizing deep learning methods in object recognition-related fields is steadily underway. In addition, research to improve recognition performance through data proliferation is also making significant progress. In particular, for the stable and good performance of object recognition technologies, it is unbiased and requires dozens to millions of vast datasets. Shared datasets, mainly available online, consist of common objects such as vehicles, people, cups, animals, etc. However, datasets on objects that can be seen in industrial or personally specialized environments are not only difficult to obtain, but also take considerable time and manpower to build.

In this paper, we present a dataset construction framework for improving the performance of object recognition in specialized environments. To verify this, performance evaluations are conducted through deep learning and accuracy is verified through testing in real-world environments. We construct a customized dataset for object detection. First, we conduct online data collection on custom

datasets that are not obtained from shared datasets, and to compensate for the insufficient amount and biased data, we conduct data proliferation by building processes consisting of data manipulation and deep learning Generative Adversarial Networks (GANs). We apply rotational, inversion, blur, and random noise as image manipulation, and deep learning-based GAN are trained via DCGAN, WGAN, and SRGAN structures. We build a data proliferation framework for optimizing efficiency and performance of these techniques.

To validate the performance of the corresponding dataset, we conduct learning with a YOLO family structure, a 1-stage object recognition algorithm. This is a compromised algorithm of accuracy and real-time recognition time, which is applied because it brings appropriate accuracy results in a short time. Deep learning learning is conducted based on the total YOLOv4, YOLOv3, and YOLOv3-tiny structures. In YOLOv4 learning, we leverage the state-of-the-art deep learning techniques, Mish activation function, DropBlock performing interrelated zone dropout, and CmBN to generate optimal models. In addition, we generate a learning model with YOLOv3, YOLOv3-tiny structure per dataset.

For performance verification, we leverage AP evaluation metrics to compute the area of PR curves containing factors from Precision and Recall. The performance was verified by extracting AP accuracy with the same test dataset for each collected dataset, proliferating dataset, and verification was conducted based on various YOLO structures. The presented multiplication dataset confirms an average accuracy improvement of 10-20% on an mAP basis and a performance improvement of up to 36% compared to the existing collected datasets.

Additionally, we conduct embedded-based object recognition performance tests in real-world environments using the generated optimal model. At this time, we leverage the platform of ROS-based UGV to verify object recognition performance. We also confirm the cognitive accuracy and speed performance on the mounted embedded boards of UGV, NANO, TX2, and Xavier. Real-time accuracy is checked by learning model around the object we want to recognize to confirm the accuracy improvement. We also provide a rate of 54 FPS based on Xavier, 24 FPS based on TX2, and 12 FPS based on NANO on YOLOv3-tiny basis. We provide Xavier-based average 14FPS, TX2-based average 8FPS, and NANO-based average 3FPS on the YOLOv3 basis.

we present a framework for building a dataset of customized objects, which we conduct deep learning with the structure of the YOLO family to generate an optimal model. The performance verification of the generated model was conducted based on AP, and the stable and high performance of the dataset presented by testing according to the dataset type in real-time environment was verified. In addition, performance evaluations are conducted using various embedded systems to improve the cognitive speed of object recognition.

Key Words : Deep learning, Object detection, Image augmentation, GAN(Generative Adversarial Network), ROS(Robot Operating System)