REPORT

Bounding Box-Based YOLO

과목[분반]	딥러닝[01]
과제명	YOLO를 사용한 테니스 선수 및 공 검출 프로젝트
담당교수	안대한
학과/학년	IT융합학부/3학년
학번	20202086 20202123
이름	권다운 오준석
제출일(년/월/일)	2024.06.26

[Contents]

1. 연구 주제 (요약)

2. 목표 (데이터셋 및 관련 연구)
2.1. 데이터셋: Roboflow Dataset
2.2. R-CNN 계열 아키텍처의 Detection Network
2.3. Bounding Box-Based YOLO를 통한 테니스 관련 객체 검출

3. 모델 설계 및 학습 방법
3.1. Leaky ReLU에서 GELU로 변경
3.2. Batch Normalization에서 Layer Normalization로 변경
3.3. Xavier 가중치 초기화
3.4. Bounding Box-Based Confidence 연산
3.5. 학습을 위한 하이퍼파라미터 세팅

4. 데모 및 평가
4.1. 공 검출에 대한 문제점 분석
4.2. YOLO vs Bounding-Box Based YOLO
4.3. Bounding Box-Based YOLO에 대한 문제점 분석

5. 느낀 점 (결론, 추후 연구)
5.1. 추후 연구: Classification의 Probability에 기반한 Confidence 연산
5.2. 프로젝트를 통해 느낀 점

[1. 연구 주제]

본 프로젝트에서는 YOLO를 사용하여 주어진 테니스 경기 영상에서 테니스 선수와 공을 검출하는 것을 목표로 한다. 실시간 영상 처리에서 R-CNN 계열의 모델보다 YOLO 계열의 모델이 주목받는 이유는 YOLO가 1-Stage Detector로 영역 제안을 수행하지 않으면서 Bounding Box와 클래스에 대한 확률을 빠르게 도출해낼 수 있기 때문이다. 하지만, R-CNN 계열의 모델은 영역 제안을 하는 과정을 포함하고, 객체를 정확하게 검출해내기 위해 상대적으로 복잡한 구조로 설계되었기 때문에 더 정확한 검출이 가능하다. 이에 따라 본 프로젝트에서는 R-CNN의 Detection Network 구조에 기반한 방식을 채택하여 Bounding Box-Based YOLO라는 모델을 구축하여 프로젝트를 수행하였다. YOLO와 제안된 모델을 비교하기 위해 Pascal VOC 2012 Dataset을 통해 mAP를 측정하여 이를 비교하였으나, **제안된 모델의 성능이 더 우수하지 않았음을 미리 알린다.** 이에 대해 문제점을 분석하고, 추후에 새로운 방향의 개선 방법을 제안하였다. 본 프로젝트의 소스코드는 Github를 통해 사용할 수 있다.
(https://github.com/drawcodeboy/BoundingBox-Based-YOLO)

[2. 목표 (데이터셋 및 관련 연구)]

2.1. 데이터셋: Roboflow Dataset

테니스 선수와 공을 검출하기 위해서 Roboflow에 업로드 되어있는 데이터셋(Fig 1)을 사용하였다. (https://universe.roboflow.com/tennistracker-dogbm/object-detection-iqd6e) 버전 6까지를 제공하고 있는 Dataset이지만, 버전 5부터는 데이터가 흑백 이미지를 제공하고 있기 때문에 컬러 이미지를 제공하는 버전 4를 본 프로젝트 Dataset으로 채택하였으며, 해당 Dataset은 1,722개의 이미지를 Train set으로 제공하고 있다. 또한, 이미지 크기가 모두 1920x1080 사이즈로 모델이 처리하기에 매우 많은 파라미터의 수(약 3억 개)를 갖게 된다. 이에 대해 크기를 960x540으로 줄여서 모델이 더 적은 파라미터의 수로(약 1.6억 개) 처리할 수 있게 하였다. 또한, 라벨링은 (Player 1, Player 2, Ball)로 구성이 되어있다.

Fig 1: Roboflow Dataset

2.2. R-CNN 계열 아키텍처의 Detection Network

Object detection Task에서는 1-Stage Detector인 YOLO와 2-Stage Detector인 R-CNN이 이를 해결하기 위한 베이스 아키텍처로 주목되어오고 있다. 하지만, YOLO는 1-Stage로 빠른 추론 속

도를 보이는 것에 반해 R-CNN보다 상대적으로 정확도는 떨어진다. R-CNN은 2-Stage로 구성되어 있으며, 1st Stage에서는 영역 제안을 통해 Bounding Box가 있을 만한 후보 영역들을 추출하는 단계와 2nd Stage에서는 이러한 후보 영역들을 기반으로 더 세밀한 Bounding Box를 검출하고, 내부에 어떠한 객체가 있는지를 분류하는 Detection Network로 구성되어 있다.

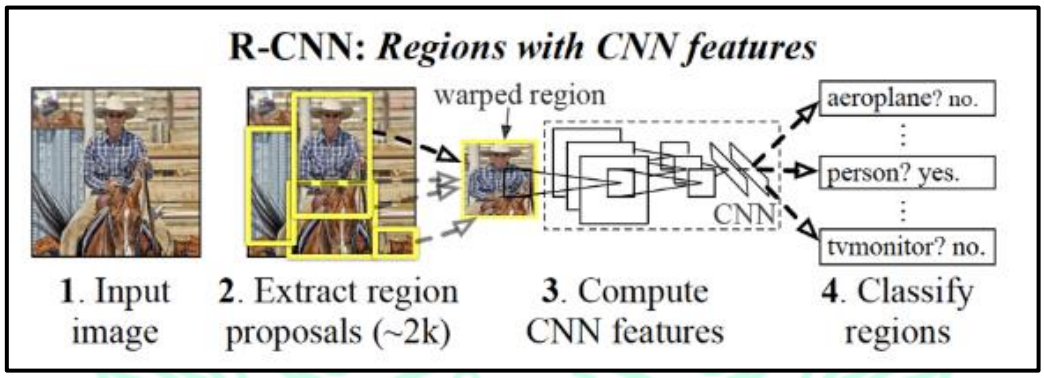

Fig 2: R-CNN Process

R-CNN의 후보 영역을 기반으로 객체를 검출하는 방식을 채택하여 본 프로젝트에서는 Bounding Box-Based YOLO라는 모델을 제안하고자 한다. 이를 위해 YOLO의 output으로 도출된 Bounding Box의 좌표 값들에 대해 1x1 Convolution을 통해 feature를 추출하고, Confidence에 영향을 끼치도록 하게끔 모델을 설계하였다. 이에 대한 자세한 설계 내용은 3.4절에서 다루도록 한다.

2.3. Bounding Box-Based YOLO를 통한 테니스 관련 객체 검출

문제와 목표를 다시 정의하면, R-CNN의 Detection Network에서 영감을 얻어 설계한 Bounding Box-Based YOLO를 통해 roboflow의 데이터셋을 학습하여 주어진 테니스 경기 영상에 대해 객체를 검출하는 것이 이 프로젝트의 목표이다. 3장에서는 본 프로젝트에서 제안하는 모델에 대한 설계를 자세하게 다루고, 4장에서는 학습이 완료된 모델에 실행시키고 영상을 추론하여 파악된 문제점들에 대한 분석을 진행하였다. 또한, 제안된 모델을 기존의 YOLO와 성능 비교를 위해서 PASCAL VOC 2012 Dataset을 통해서 같은 조건의 학습을 진행시켜서 mAP를 측정하고, 낮은 성능에 대한 원인을 분석하였다. 마지막으로, 5장에서는 이러한 문제점들을 기반으로 추후에 진행해야 하는 새로운 모델의 설계에 대한 계획으로 마무리한다.

[3. 모델 설계 및 학습 방법]

3.1~3.3절에는 기존 YOLO에서 사용되는 Darknet의 CBA Block에 대한 수정 사항에 대해 작성되었으며, 3.4절에는 Confidence를 추출하는 방식, 3.5절에서는 학습을 위한 하이퍼파라미터 세팅 값들을 작성하였다.

3.1. Leaky ReLU에서 GELU로 변경

GELU는 입력 값이 0에 가까울 때도 작은 기울기를 유지하여 학습 초기 단계에서 Gradient Vanishing 문제를 완화시킬 수 있다. 반면에 Leaky ReLU는 음수 영역에서 일정한 기울기를 가지지만, 이로 인해 일부 입력 값에서 비효율적인 학습이 발생할 수 있기 때문에 기존 YOLO Darknet의 활성화 함수를 GELU로 채택하였다.

3.2. Batch Normalization에서 Layer Normalization으로 변경

Batch Normalization은 미니배치 크기에 대해 의존적이어서, 작은 배치 크기에는 성능이 저하될 수 있다. 본 프로젝트에서는 원래 미니매치 크기를 32로 고려하였으나, Colab에서 할당할 수 있는 최대의 메모리 크기를 넘어서서 오류가 발생하였기 때문에 16으로 크기를 바꾸게 되었다. 이에 대해 배치 크기에 의존하지 않도록 Layer Normalization으로 CBA Block을 수정하였다. 또한, 이외에도 입력 데이터의 분포 변화에 대해 덜 민감하다는 이점이 있다.

3.3. Xavier 가중치 초기화

Xavier 가중치 초기화는 입력과 출력의 분산을 고려하여 가중치를 초기화함으로써, 입력 신호가 각 층을 통과할 때 과도하게 증폭되거나 감소하지 않도록 한다. 이는 Gradient Vanishing, Exploding 문제를 완화시켜 학습을 안정되게 한다. 또한, 초기화된 가중치가 적절한 값으로 설정되기 때문에 학습 초기에 모델이 더 빠르게 수렴할 수 있다. 이는 학습 속도를 가속화하고, 최적의 성능에 도달하는 데에 필요한 시간을 단축시킨다. 하지만, 빠른 수렴이라는 이점은 훈련을 시킬 때, loss 값을 비교한 결과 보이지 않은 것으로 판단된다. (그림 #) 이에 대한 구현은 PyTorch에서 nn.Module.apply() 메서드를 통해 구현할 수 있었다.

3.4. Bounding Box-Based Confidence 연산

3.4.1. MLP를 통한 Bounding Box 좌표 값의 특성 추출

기존 YOLO의 Confidence는 R-CNN과 달리 Bounding Box, Classification Probability와 독립적으로 입력 이미지로부터 값을 추출한다. 이에 대해 R-CNN과 유사하게 본 프로젝트에서는 각 셀의 Bounding Box 결과 값에서 MLP를 통해 feature를 추출하도록 하였다. 하지만, 하나의 Linear Layer로 추출하기에는 좌표 값에 대한 정보가 유지될 수 있기 때문에 Linear Layer를 2개로 구성하여 Fig 3과 같이 구성하였다.

추출된 feature에 대해 Sigmoid 함수를 통해 하나의 Probability 값으로 나타낸다. Probability 값으로 나타낸 이유는 (0, 1) 사이의 값으로 나타내고, 해당 값을 Confidence에 곱셈 연산을 하여 Confidence를 추출할 때, 해당 Bounding Box 내에 객체가 있을 확률에 대한 신중함을 나타내기 위함이다.

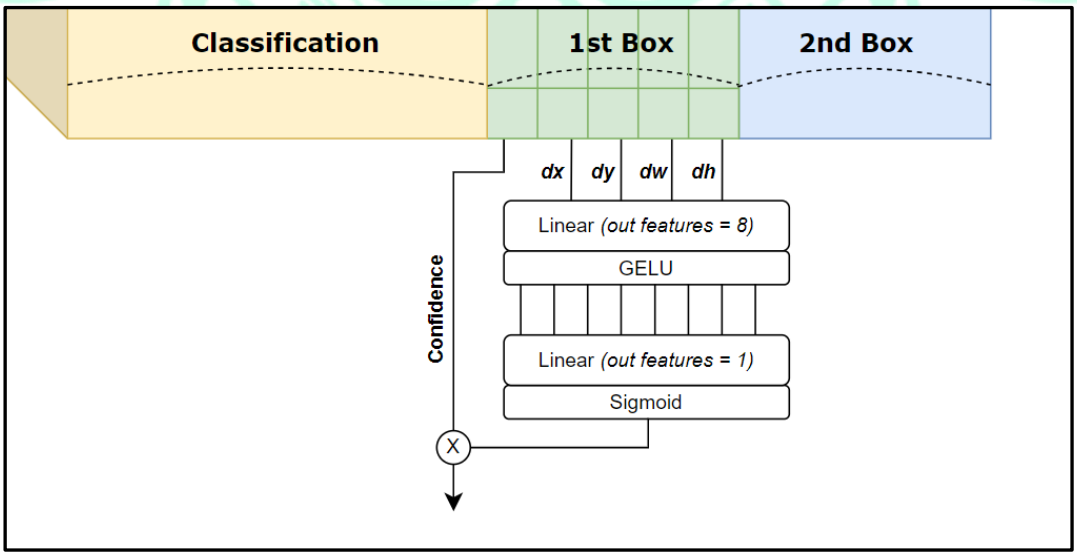

Fig 3: Bounding Box-Based Confidence (each cell, and each box)

3.4.2. 1x1 Convolution

하지만, 각 박스에 각 셀에 대해 MLP로 설계할 경우에 구현상에서 병렬적으로 처리를 하지 못한다는 단점이 존재했다. 실제로 1x1 Convolution을 적용한다고 해도, 슬라이딩 윈도우 방식으로 작동하기 때문에 완전한 문제를 해결하지 못하지만, 1x1 Convolution을 적용할 경우에 각 셀에 있던 모든 MLP 파라미터가 1x1 Conv의 파라미터로 전부 압축되어 파라미터의 수를 줄일 수 있다는 이점이 있다. 이에 기반하여 각 셀의 MLP를 1x1 Conv Layer로 변경하였다.

3.5. 학습을 위한 하이퍼파라미터 세팅

테니스 검출을 위한 하이퍼파라미터 세팅과 YOLO와 Bounding Box-Based YOLO를 비교하기 위해 PASCAL VOC 2012 Dataset을 학습할 때의 세팅 값은 모두 (Table 1)과 동일하다.

Batch Size	*Learning Rate*	*Optimizer*	*Epochs*
16	1e-5	Adam	100

Table 1: Hyperparameter setting

[4. 데모 및 평가]

(데모는 모델의 가중치 pt 파일이 매우 크기 때문에 제출한 영상으로 대체하였다.)

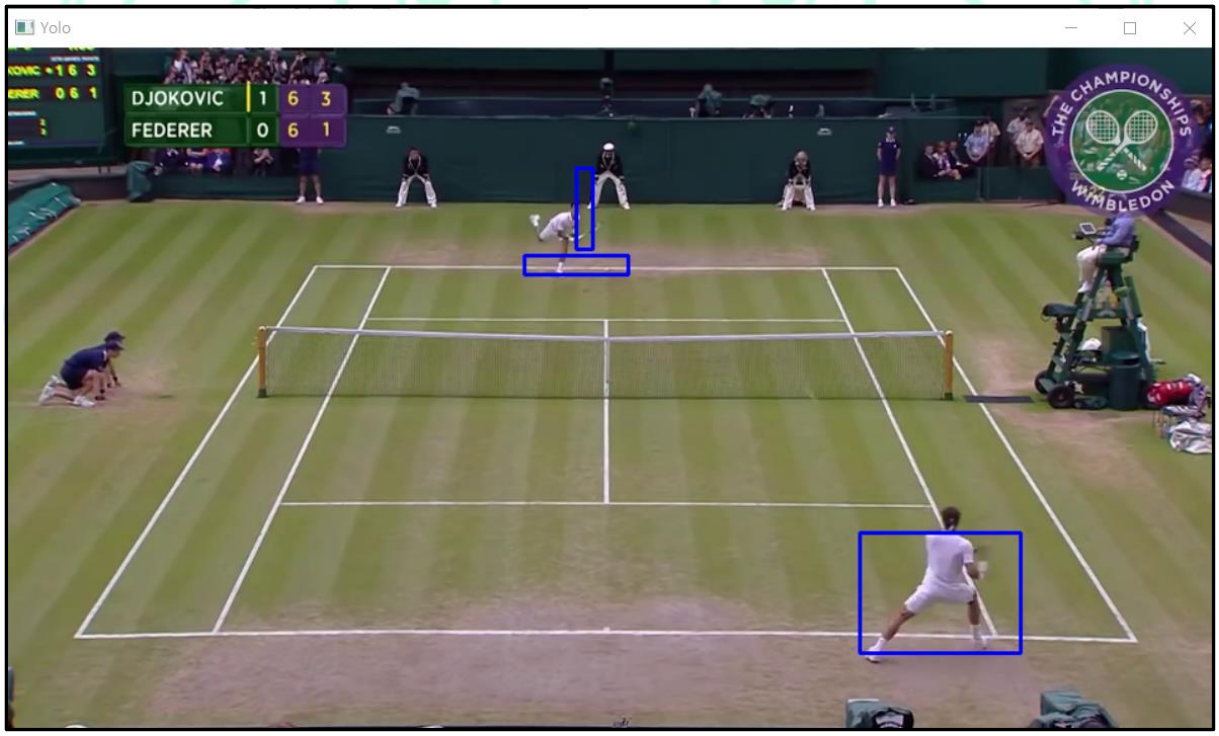

Fig 4: Bounding Box-Based YOLO inference Tennis Video

4.1. 공 검출에 대한 문제점 분석

선수에 대한 객체는 잘 검출하지만, 공에 대해서는 거의 검출을 해내지 못한다. 데이터셋에는 공에

대한 라벨링이 충분히 되어있으나 잘 검출해내지 못하는 것은 다음과 같은 이유들로 추측해볼 수 있었다.

1) 공의 크기가 너무 작다.
2) 데이터셋은 경기장의 색이 공의 색과 서로 다른 것이 대부분인 것에 반하여, 영상은 공과 경기장의 색이 거의 유사하다.

이러한 문제점에 대해서 학습이나 추론할 데이터를 모두 흑백 이미지로 처리하거나 Inception에서 제안한 여러 객체 크기를 고려한 Inception Module을 채택하는 것도 해결 방법이 될 수 있을 것으로 추측된다.

4.2. YOLO vs Bounding Box-Based YOLO

제안된 모델과 YOLO의 성능을 비교해야 한다. 이를 위해 PASCAL VOC 2012 Dataset을 사용했으며, Table 1과 같은 하이퍼파라미터 세팅을 통해 학습을 하고, mAP를 통해 성능을 평가했다. 이미지의 크기는 모두 448x448로 조절하였으며, 학습 이미지의 수는 13,690개, 테스트 이미지의 수는 3,422개이다. mAP를 측정하기 앞서 훈련 간의 epoch별로 loss를 측정하였다(Fig 5). 미세하게 제안된 모델의 loss 값이 더 낮지만, 이것으로 성능 부분에서 판단을 하기에는 어렵다.

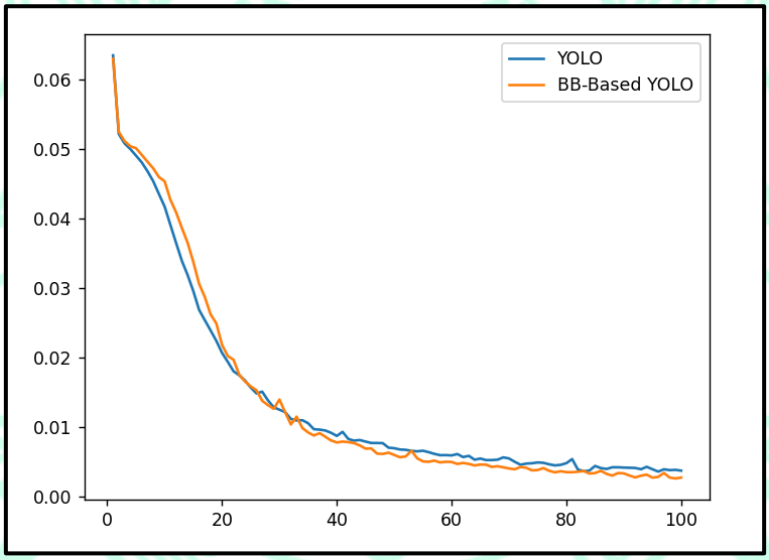

Fig 5: YOLO loss vs Bounding Box-Based YOLO loss

두 모델에 대해 IoU Threshold에 따른 mAP를 측정한 결과 Table 2와 같다.

	mAP(IoU>=0.3)	mAP(IoU>=0.5)	mAP(IoU>=0.7)
YOLO	0.363	**0.233**	**0.060**
Proposed Model	**0.402**	0.223	0.019

Table 2: YOLO vs Bounding Box-Based YOLO mAP

4.3. Bounding Box-Based YOLO에 대한 문제점 분석

제안한 모델에 대해 가장 큰 문제점은 2가지로 추측이 된다. 1번째는 기존 Confidence에 대한 곱셈 연산이다. 신중한 Confidence 값을 도출하기 위해 곱셈 연산을 통해 낮추었던 값이 되려 mAP를 측정할 때는 더 낮은 값을 도출하게 되는 것이 원인으로 분석된다. 2번째는 Bounding Box를 기반한 feature의 추출이다. Bounding Box 자체는 좌표 값으로 RoI에 대한 이미지 정보가 아닌 위치 정보

이다. 이에 대한 feature를 추출하여 새로운 정보를 더해주려 한 것은 모순이며, 큰 의미가 없는 것으로 추측된다.

[5. 느낀 점 (결론, 추후 연구)]

5.1. 추후 연구: Classification의 Probability에 기반한 Confidence 연산

프로젝트가 끝난 직후에 문제점을 분석하면서, Bounding Box가 아닌 Classification의 확률 값 정보를 활용하는 것이 더 논리적인 방법이 될 수 있는 가능성을 보았다. 확률 값이 높을수록 객체가 어떤 정보를 가지고 있는가에 대한 확신으로 확률 값에 대해 특정 상수(0.1, 0.01, 0.001, ...)를 곱해준 뒤에 이번엔 곱셈 연산이 아닌 덧셈 연산을 수행한다면, 좋은 결과를 도출할 수 있을 것으로 예상된다.

5.1. 프로젝트를 통해 느낀 점

다소 짧은 5일이라는 시간 안에 아이디어, 구현, 학습, 평가, 레포트 작성이 이루어져 면밀한 작업들이 수행되지 않았던 부분들이 아쉽게 느껴진다. 하지만, 아이디어를 모델에 적용함으로써 YOLO에 대한 이해도를 높이고, Object Detection이라는 Task에 대해 한 층 더 깊게 고민할 수 있게 된 부분들에 있어서 만족도를 느낄 수 있는 프로젝트였다.