**P2 기법 적용 분석 보고서**

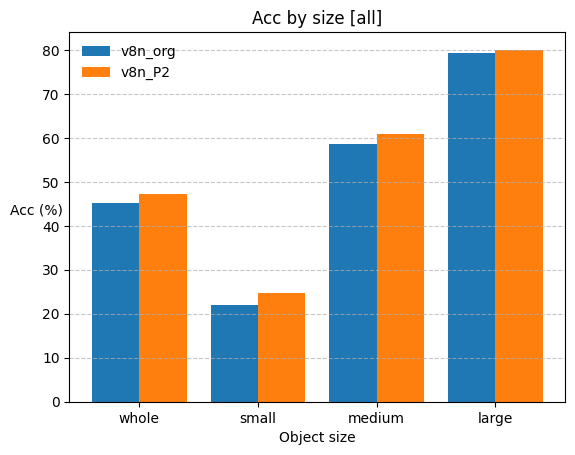
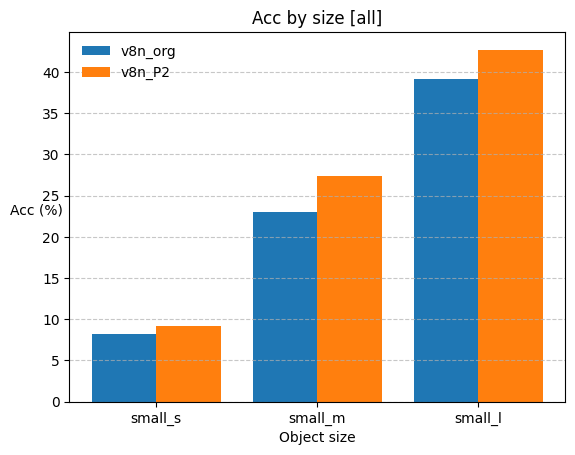
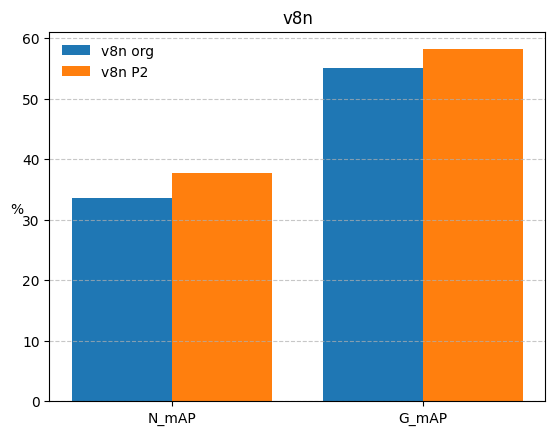
1. **서론**
   1. **배경**
      1. Feature map  
         : YOLO 모델에서 Backbone은 객체의 정보를 담는 feature map을 생성 및 추출하고 Neck에서 이를 조합하여 최종 feature map을 만들고 Head의 Detect 단으로 보낸다. 기존의 구조에서는 해상도의 크기가 각각 (80, 80), (40, 40), (20, 20) 인 P3, P4, P5 feature map을 생성 및 추출하여 사용한다.   
          객체 크기 관점에서 보면, Backbone에서 추출한 feature map의 해상도가 클수록 소형 객체 탐지에 유리한 정보들이 있고, 해상도가 작을수록 대형 객체 탐지에 유리한 정보들이 있다.
      2. P2 feature map과 소형 객체 탐지의 상관관계  
         : 소형 객체 탐지 성능을 향상시키기 위해서는 소형 객체의 정보가 담긴 feature map이 필요하다. 따라서 기존 모델에서 해상도가 (160, 160)인 P2 feature map을 적용하여 소형 객체 탐지 성능이 향상되는지 분석하는 실험을 진행한다.
   2. **실험 준비** 
      1. 실험 모델
         1. v8n\_org
         2. v8n\_P2
         3. v8s\_org
         4. v8s\_P2
   3. **목적 : P2 적용 시 소형 객체 탐지 성능 변화 분석**
      1. P2 적용시 성능 변화 분석.  
         : [모델1]을 기준으로 [모델2]를 비교하여 먼저 n모델에서 P2를 적용하였을 때 객체 탐지 성능의 변화를 관찰하고, 이후 [모델3]을 기준으로 [모델4]를 비교하여 s모델에서도 객체 탐지 성능 변화를 관찰한다.
   4. **평가 지표**
      1. 모델 성능 : GPU mAP, NPU mAP, FPS
      2. 객체 크기별 정확도
2. **본론**
   1. **모델 설계**
      1. 데이터셋
         1. 각 모델들이 학습에 사용된 데이터와 테스트에 사용된 데이터는 동일
            1. 학습 데이터 : 제공받은 train set
            2. 입력 데이터 : 제공받은 test set
      2. P2 기법 적용
      3. 실험 설계

\* org는 기본 YOLO 모델(original)  
\* P2는 기본 모델에 P2를 적용한 모델

* + - 1. v8n\_org : v8n 기본 모델
      2. v8n\_P2 : P2를 적용한 v8n 모델
      3. v8s\_org : v8s 기본 모델
      4. v8s\_P2 : P2를 적용한 v8s 모델
  1. **실험 결과 및 분석**
     1. 실험 1 P2 적용에 따른 성능 변화 분석
        1. 실험 항목
           1. 대조군1 : v8n\_org / 실험군1 : v8n\_P2
           2. 대조군1 : v8s\_org / 실험군1 : v8s\_P2
        2. **가설 – P2 적용시 소형 객체 탐지 성능이 증가할 것이다.**: [배경 i]에서 설명했듯이 해상도가 큰 feature map은 소형 객체 탐지에 유리한 정보를 가지고 있다. 따라서 기존 모델에서 해상도가 큰 P2 feature map을 적용하면 소형 객체에 대한 정보가 추가되므로 소형 객체 탐지의 성능이 향상할 것이다.

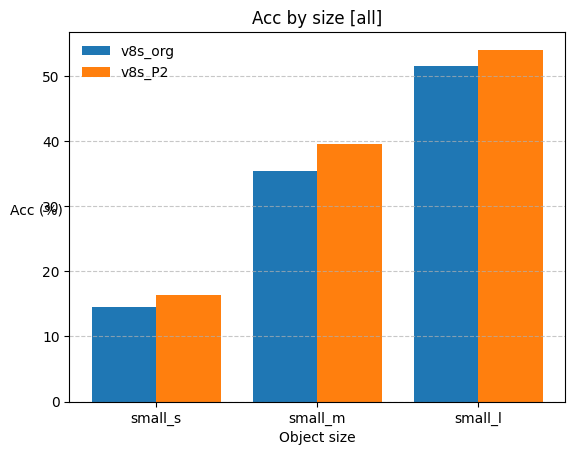
* + - 1. 결과

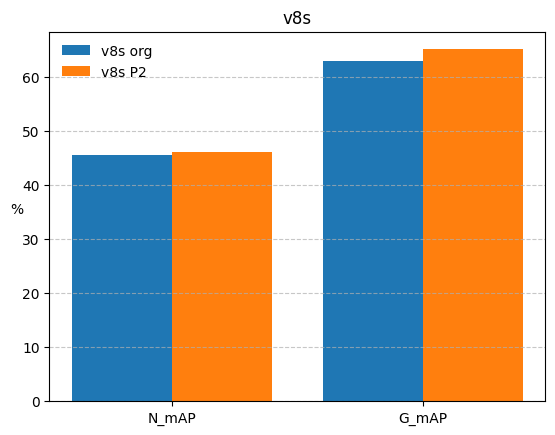
|  |  |  |
| --- | --- | --- |
|  | v8n\_org | **v8n\_P2** |
| GPU mAP (%) | 55.02 | **58.13** |
| NPU mAP (%) | 33.54 | **37.63** |
| FPS | 180.77 | **140.53** |

* + - * 1. v8n\_org vs v8n\_P2  
           - 평가 지표 : 모델 성능  
           [그래프 1]  
             
           - 평가 지표 : 객체 크기별 정확도  
           [그래프 2] [그래프 3]

: 위의 그래프와 표는 v8n\_org, v8n\_P2 모델의 성능과 객체 크기별 정확도를 나타낸 것이다.  
 먼저 모델의 성능 관점에서 분석했을 때, [그래프 1]에서 v8n\_P2 모델이 FPS는 감소했지만 GPU, NPU mAP 성능은 P2를 적용했을 때 상승하는 것을 확인할 수 있다.  
 객체 크기별 정확도 관점에서 분석했을 때, [그래프 2]에서 소형, 중형 객체에서 특히 성능이 증가하는 것을 알 수 있으며 그에 따라 전체 성능도 증가하는 것을 확인할 수 있다. 소형 객체에서의 성능을 더 자세히 파악하기 위해 [그래프 3]을 확인했을 때, small\_s에 대한 성능 향상은 크지 않지만 small\_m, small\_l 객체에 대한 성능은 크게 증가했다.  
 따라서 v8n 모델의 경우에는 v8n\_P2 모델이 FPS를 제외한 모든 부분에서 우수하다. 이후 v8s 모델에서도 같은 경향성을 보이는지 확인하기 위해 [실험 B]를 진행했다.

|  |  |  |
| --- | --- | --- |
|  | v8s\_org | **v8s\_P2** |
| GPU mAP (%) | 62.93 | **65.09** |
| NPU mAP (%) | 45.53 | **46.14** |
| FPS | 63.74 | **50.12** |

* + - * 1. 텍스트, 스크린샷, 도표, 라인이(가) 표시된 사진

           자동 생성된 설명v8s\_org vs v8s\_P2  
           - 평가 지표 : 모델 성능  
           [그래프 1]  
             
           - 평가 지표 : 객체 크기별 정확도  
           [그래프 2] [그래프 3]

: 위의 그래프는 v8s\_org 모델과 v8s\_P2 모델의 성능과 객체 크기별 정확도를 나타낸 것이다.  
 먼저 모델의 성능 관점에서 분석했을 때, [그래프 1]에서 v8s\_P2 모델이 GPU mAP는 상승했지만 NPU mAP는 약간 상승한 것을 확인할 수 있었다. n모델에 비해 상승폭이 적어진 것은 전체적인 성능 자체가 높아졌기 때문으로 판단했다.  
 이후 객체 크기별 정확도를 비교했을 때에는, [그래프 2]에서 medium, large 객체에서는 차이가 없었지만 small 객체에 대해서는 성능이 증가한 것을 확인할 수 있었다. 더 자세한 분석을 위해 [그래프 3]을 보면 small\_s, small\_m, small\_l 객체 모두 성능이 상승했다. 하지만 v8s\_P2 모델의 경우 FPS가 50.12로 하한선인 50에 거의 근접했다는 문제점을 발견했다.

1. **결론**
   1. **주요 결론**
      1. 실험 1 : P2 적용에 따른 성능 변화 분석  
         : 실험 1에서 s, n 모델 모두 P2를 적용한 모델이 기본 YOLO 모델보다 전체 성능이 증가했으며, 특히 소형 객체 탐지 성능이 크게 증가했다. 따라서 P2 feature map을 적용하는 것이 본 프로젝트의 주 목표인 소형 객체 탐지 성능 향상에 부합하며 그에 따라 모델의 전체 성능 역시 증가한다는 결론을 얻었다.
   2. **향후 연구 방향**
      1. 실험 계획
         1. 소형 객체 탐지 성능을 높이기 위해서 P2를 적용하는 것이 유리하다는 결론을 얻었기 때문에 이후의 모델들은 P2를 적용하여 실험을 진행한다. 하지만 위의 실험에서 v8s\_P2 모델의 경우 FPS가 하한선인 50에 근접했기 때문에, 이후 과정에서 경량화가 필수적이다. 따라서 v8s\_P2 모델을 기반으로 성능을 증가 시키면서 FPS는 유지하거나 소폭 상승시키는 방향을 모색한다.