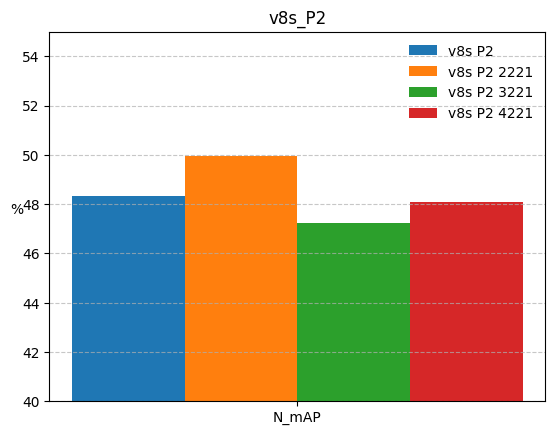
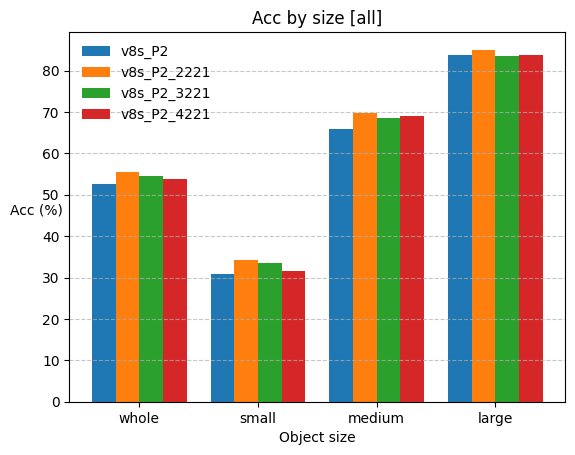
**P2 적용 및 Backbone 구조 변경 분석 보고서**

1. **서론**
   1. **배경**
      1. 소형 객체 탐지 성능 향상  
         : **[1.2 P2 기법 적용 분석 보고서]**에서 얻은 결론에 따라 P2를 적용해 본 프로젝트의 주요 목표인 소형 객체에 대한 성능을 특히 향상시키고자 한다.
      2. Backbone 구조 변경을 통한 성능 향상  
         : **[1.3 Backbone 구조 변경 분석 보고서]**에서 얻은 결론에 따라 Backbone의 Bottleneck 반복 회수를 증가시켜 모델의 성능을 개선하고자 한다.
      3. P2를 적용한 Backbone의 구조 분석  
         : **[배경 i]**, **[배경 ii]**에 따라, 본 실험에서는 Yolo v8s 모델에 P2를 적용한 v8s\_P2 모델을 기반으로 Backbone에서 소형 객체의 정보를 주로 추출하는 P2와 P3단의 반복 횟수를 조절하여 모델의 성능을 향상시키려 한다. 따라서 본 실험에서는 이전 실험들에서 얻은 두가지 기법이 동시에 적용되었을 때에도 같은 경향성이 유지되는지 확인하는 동시에 모델의 성능을 최적화하고자 한다.
   2. **실험 준비**
      1. 실험 모델
         1. v8s\_P2
         2. v8s\_P2\_1321
         3. v8s\_P2\_2221
         4. v8s\_P2\_3221
         5. v8s\_P2\_3321
         6. v8s\_P2\_4221

* 1. **목적**
     1. 실험 1 : P2단 반복 횟수에 따른 성능 변화 분석  
        : [모델 1]을 기준으로 P2단의 반복 횟수만 증가시킨 [모델 3], [모델 4], [모델 6]을 비교한 결과를 통해 P2단의 반복 횟수에 따른 모델의 성능을 관찰한다. 이 때 결과는 본 실험의 목적에 부합하게 모델 성능과 객체 크기별 정확도 두가지 관점에서 분석한다.
     2. 실험 2 : P3단 반복 횟수에 따른 성능 변화 분석  
        : [모델 1]을 기준으로 P3단의 반복 횟수만 증가시킨 [모델 2]를 비교한 결과와 [모델 4]를 기준으로 마찬가지인 [모델 5]를 비교한 결과를 종합해 P3 단의 반복 횟수에 따른 모델의 성능을 관찰한다. 이때 결과는 마찬가지로 모델 성능과 객체 크기별 정확도 두가지 관점에서 분석한다.
  2. **평가 지표**
     1. 모델 성능 : GPU mAP, NPU mAP, FPS
     2. 객체 크기별 정확도

1. **본론**
   1. **실험 설계**
      1. 데이터셋
         1. 각 모델들이 학습에 사용된 데이터와 테스트에 사용된 데이터는 동일
            1. 학습 데이터 : 제공받은 train set과 validation set을 병합
            2. 테스트 데이터 : 제공받은 test set
      2. 모델 설계

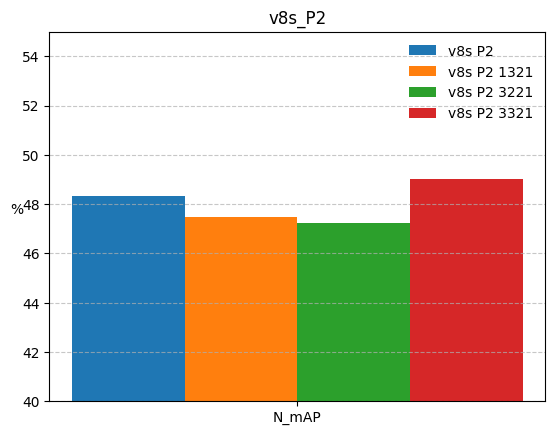
\* 모델명 뒤 1321 ~ 4221는 Backbone에서 C2f 모듈 내의 Bottleneck 반복 횟수  
 본 실험에 사용된 모델은 모두 P2 적용

* + - 1. v8s\_P2 : v8s 모델에 P2를 적용한 모델, Bottleneck 반복 횟수는 (1, 2, 2, 1)
      2. v8s\_P2\_1321 : Bottleneck 반복 횟수 (1, 3, 2, 1)
      3. v8s\_P2\_2221 : Bottleneck 반복 횟수 (2, 2, 2, 1)
      4. v8s\_P2\_3221 : Bottleneck 반복 횟수 (3, 2, 2, 1)
      5. v8s\_P2\_3321 : Bottleneck 반복 횟수 (3, 3, 2, 1)
      6. v8s\_P2\_4221 : Bottleneck 반복 횟수 (4, 2, 2, 1)
  1. **실험 결과 및 분석**
     1. 실험 1 : P2단 반복 횟수에 따른 성능 변화 분석
        1. 실험 항목
           1. 대조군1 : v8s\_P2 (1221)  
              실험군1 : v8s\_P2\_2221, 실험군2 : v8s\_P2\_3221, 실험군3 : v8s\_P2\_3221  
              : v8s\_P2 모델에서 나머지는 그대로 두고 실험군들은 P2단의 반복 횟수만 증가시켜 변인통제를 하였다.
        2. **가설 – P2단의 반복 횟수가 증가하면 성능이 증가할 것이다.**: **[1.2 P2 기법 적용 분석 보고서]**에서 P2 Layer를 추가했을 때 모델의 성능이 증가함을 확인했다. 또한 **[1.3 Backbone 구조 변경 분석 보고서]**에서 Backbone의 Bottleneck 반복 횟수를 적절히 증가시키면 모델의 성능이 증가함을 확인했다. 따라서 소형 객체에 대한 정보를 추출하는 P2단의 반복 횟수를 늘리면 소형 객체에 대한 성능이 주로 증가하며 전체 성능 역시 증가할 것이라고 가정했다.
        3. 결과  
           - 평가 기준 : 모델 성능, 객체 크기별 정확도  
           [그래프 1] [그래프 2]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | v8s\_P2 (1221) | **v8s\_P2\_2221** | v8s\_3221 | v8s\_4221 |
| GPU mAP (%) | 67.82 | **68.74** | 68.88 | 68.98 |
| NPU mAP (%) | 48.34 | **49.94** | 47.24 | 48.07 |
| FPS | 51.03 | **49.69** | 46.78 | 44.84 |

* + - 1. 분석  
         : 위 그래프는 동일 조건에서 P2단 반복 횟수에 따른 모델의 성능, 객체 크기별 정확도를 나타낸다.  
          먼저 모델의 성능 관점에서 분석하기 위해 [그래프 1]을 확인했을 때, v8s\_P2\_2221 모델이 가장 높게 나왔다. 그리고 1221 -> 2221의 경우, 3221 -> 4221의 경우 모두 가설과 같이 N\_mAP의 증가를 확인할 수 있었다. 하지만 2221 -> 3221의 경우 성능이 큰 폭으로 감소하는 것을 발견했다.  
          객체 크기별 정확도 관점에서 분석하기 위해 [그래프 2]를 확인했을 때에는, 1221 -> 2221의 경우 small, medium 객체에서 크게 향상되어 전체 성능 역시 개선된 것을 알 수 있었다. 하지만 2221 -> 3221 -> 4221의 경우 성능이 오히려 하락하는 것을 발견했다.  
          이에 대한 원인으로는 P2단의 반복수가 나머지들보다 커지는 경우에는 오히려 다른 객체들에 대한 정보량이 줄어들어 모델의 성능이 감소한 것이라고 판단할 수 있다.
    1. 실험 2 : P3단 반복 횟수에 따른 성능 변화 분석
       1. 실험 항목
          1. 대조군1 : v8s\_P2, 실험군1 : v8s\_P2\_1321
          2. 대조군2 : v8s\_P2\_3221, 실험군2 : v8s\_P2\_3321  
             : 대조군에서 P2, P4, P5단의 반복 횟수는 동일한 상태에서 실험군은 P3단의 반복 횟수만 증가시켜 변인통제 하였다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | v8s\_P2 (1221) | v8s\_P2 1321 | v8s\_P2 3221 | v8s\_P2 3321 |
| NPU mAP (%) | 48.34 | 47.49 | 47.24 | **49.04** |
| FPS | 51.03 | 50.05 | 46.78 | 45.26 |

* + - 1. 결과  
         - 평가 기준 : 모델 성능 (NPU mAP, FPS)  
          [그래프 1]
      2. 분석  
         : 위 그래프는 동일 조건에서 P3단의 반복 회수를 늘렸을 때에 대한 모델 성능과 객체 크기별 정확도를 나타낸다.  
          모델의 성능 관점에서는 v8s\_3321이 가장 우수했지만 FPS가 약 45로 기준에 미치지 못하였다. 실험 A에서는 1221 -> 1321로 변화하면서 모델의 성능이 감소했지만, 실험 B에서는 3221 -> 3321로 변화하면서 모델의 성능이 큰 폭으로 증가했다.  
          실험 A의 결과는 P3에서 추출되는 정보가 비중이 과하게 높아져 오히려 성능을 하락시킨 것으로 해석할 수 있다.  
          그리고 실험 B의 결과는 P2의 정보량이 과하게 높은 3221 에서 3321로 증가하는 과정에서 P2, P3, P4, P5단에서 추출되는 정보량의 비율이 오히려 적절하게 이루어졌기 때문에 큰 폭의 성능 향상이 있는 것으로 볼 수 있다.  
          따라서 P3단에   
            
           
           
         다만, 모듈 간의 Bottleneck 반복 횟수의 Balance를 적절하게 맞춰주어 조정했을 때 NPU 성능이 증가함을 확인했다.  
         따라서, P2~P5 단의 반복 횟수 간 편차를 크지 않게 Balance를 적절히 맞춰야 하는 인사이트를 도출했다.  
           
         [실험항목 C]에서는 두 모델 모두 Bottleneck 반복 횟수의 편차가 고르지 않지만 v8s\_P2\_3321 모델이 P2, P3 단의 반복 횟수를 균일하게 맞춤으로써 v8s\_P2\_3221 모델보다 편차가 고르기 때문에 NPU 성능이 좋게 나왔을 것이다.

1. **결론**
   1. **주요 결론**
      1. 실험 1 P2를 추가한 후 Backbone에서의 Bottleneck 반복 횟수 조정에 따른 성능 변화 분석  
         : 소형 객체 탐지에 유리한 P3 단의 Bottleneck 반복 횟수만 늘릴 경우 오히려 NPU 성능이 감소했다. P2~P5 단의 Bottleneck 반복 횟수를 균형 있게 조정하였을 때 NPU 성능이 가장 높았다.  
         따라서, [분석 보고서 1.3] 결론에 언급한 바와 같이 사용할 모델의 종류와 Scale에 따라 Bottleneck 반복 횟수를 균형 있게 조합해야 높은 성능을 얻을 수 있다.
   2. **향후 연구 방향**
      1. 향후 실험  
         추가 실험을 통해 v8s 모델에서 Bottleneck 반복 횟수를 균형 있게 맞추어 최적의 모델을 탐색한다.  
         또한, 균형 있는 Bottleneck 반복 횟수를 가지더라도 오히려 과도하게 늘릴 경우 발생할 수 있는 성능 저하에 대한 연구 방향을 설정한다.