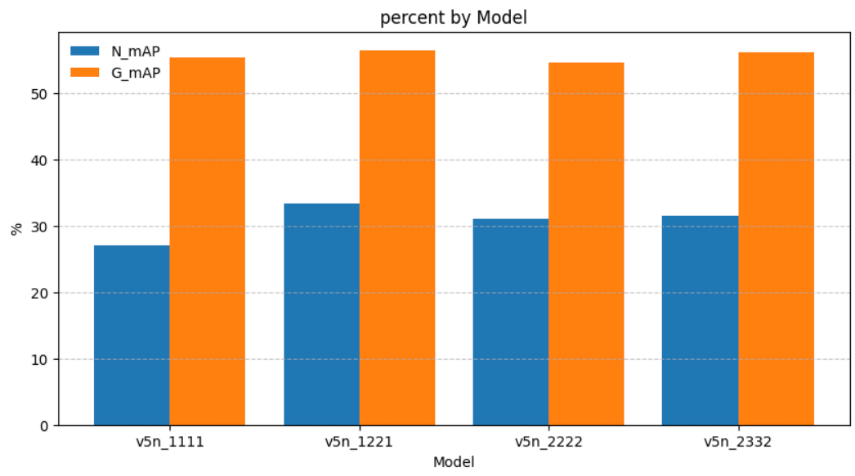
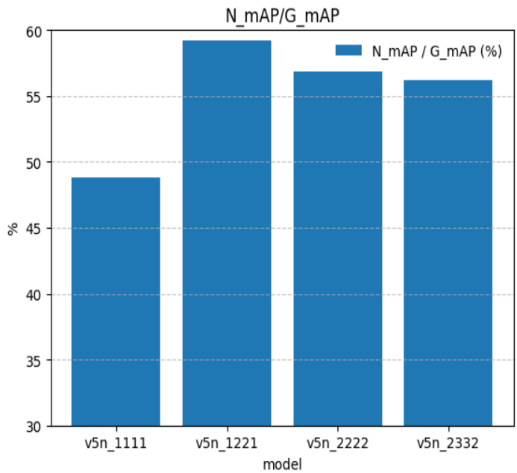
**Bottleneck 반복 수와 보존율 분석 보고서**

1. **서론**
   1. **배경**
      1. Yolo 모델의 구조 형성 방법  
         : 모델의 구조를 선언하는 yolo.yaml에서 모델의 사이즈를 결정하는 [depth, width, max\_channel] 인자가 있으며, 각 모듈이 어떻게 적용되는지에 대해 선언한 [from, number, module, args] 형태의 인자가 있다. 이 중 C2f, C3 모듈에 대해서 둘 모두 내부에 Bottleneck 모듈이 존재하는데,  
         [ Bottleneck 반복 수 = number \* depth(s, n 모델 기준: 0.33) ]  
         의 형태로 반복 수가 결정된다.
      2. C3 모듈의 동작 방식  
         :   
         을 받아 c3 모듈에서의 Bottleneck 반복 수가 결정된다. c3 모듈의 동작방식은 먼저 입력 갈래를 두 갈래로 나눈 후 첫 번째 갈래는 cv1과 Bottleneck 블록을 통과한다. 두 번째 갈래는 cv2를 통해 전달되며, 원본정보를 보존한다. 그 후 두 경로의 출력값을 채널 방향으로 결합하고 최종 cv3를 거쳐 출력한다. 본 보고서는 v5n기준 Bottleneck 반복수에 따른 보존율의 상관관계를 분석할 예정이다.
   2. **실험 준비**
      1. 실험 항목
         1. v5n\_1111
         2. v5n\_1221
         3. v5n\_2222
         4. v5n\_2332
   3. **목적**

보존율 = NmAP / GmAP

* + 1. **Bottleneck 반복 수와 보존율의 상관관계 분석**  
       : Bottleneck 반복이 많아질수록 레이어에서의 중간 표현이 더 복잡하다. 양자화가 복잡한 표현을 단순 정수 범위로 표현해야 하므로 반복수가 많으면 오차가 커져 보존율이 하락할 수 있다. 본 실험의 목적은 적절한 Bottleneck 반복 수를 최적화하기 위해 각 모델의 보존율을 관찰한다.
  1. **평가 지표**
     1. Parameter 수
     2. NPU/GPU 비율
     3. NPU 성능
     4. GPU 성능
     5. FPS

1. **본론**
   1. **실험 설계**
      1. 실험 방법
         1. 동일한 데이터로 학습한 모델에 대해 동일한 입력 데이터를 사용하여 실험 모델 별 성능 측정.
            1. 학습 데이터 : 제공받은 train set과 추가로 add set을 병합
            2. 입력 데이터 : 제공받은 test set
         2. 실험할 주제 외의 항목은 통제.
   2. **실험 결과 및 분석**
      1. Bottleneck 반복 수에 따른 보존율 비교  
         1. 비교 모델 :
            1. v5n\_1111 vs v5n\_1221 vs v5n\_2222 vs v5n\_2332
         2. 모델 설명 :
            1. v5n\_1111: 기본 yolov5n 모델에 bottleneck 반복 수 1, 1, 1, 1로 설정한 모델
            2. v5n\_1221: 기본 yolov5n
            3. v5n\_2222: 기본 yolov5n 모델에 bottleneck 반복 수 2, 2, 2, 2로 설정한 모델
            4. v5n\_2332: 기본 yolov5n 모델에 bottleneck 반복 수 2, 3, 3, 2로 설정한 모델
         3. **가설** **– Bottleneck 반복 수가 늘어나면 보존율이 하락할 것이다.**  
             **Bottleneck** 반복 수가 증가하면 모델의 연산 복잡도가 늘어나고 레이어 간 연결과 정보의 흐름이 깊어진다. 이는 8bit 연산에서 양자화 손실을 증가시킬 가능성이 있다[1]. 따라서 1111 모델에서 Bottleneck 반복수가 증가한 모델일수록 보존율 하락폭이 클 것이다.
         4. 결과 :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Parameter 수 | **NPU/GPU 비율** | NPU 성능 | GPU 성능 | FPS |
| v5n\_1111 | 2.41M | **48.81%** | 27.4% | 55.4% | 218.11 |
| v5n\_1221 | 2.46M | **59.21%** | 33.43% | 56.46% | 204.52 |
| v5n\_2222 | 2.31M | **56.82%** | 31% | 54.56% | 206.18 |
| v5n\_2332 | 2.68M | **56.22%** | 31.56% | 56.14% | 190.07 |

* + - 1. 분석 :
         1. **v5n\_1111 -> v5n\_1221**

보존율 기준 48.81%에서 59.21로 약 10.4% 상승폭을 보인다. NPU는 Bottleneck 반복 수가 늘어나면 더 효율적으로 병렬 연산을 수행하여 성능 개선에 크게 기여한다.

* + - * 1. **v5n\_1221 -> v5n\_2222**

보존율 기준 59.21%에서 56.82%로 약 2.4% 감소폭을 보인다. Bottleneck 반복수가 더 증가함에 따라 NPU에서의 연산 처리 효율이 감소한다.

* + - * 1. **v5n\_2222 -> v5n\_2332**

보존율 기준 56.82%에서 56.22%로 0.6%의 추가 감소폭을 보인다. 과도한 Bottleneck의 반복은 NPU에서의 적정 연산량보다 연산량이 포화되어 성능 개선이 거의 없거나 저하를 초래한다. Bottleneck 반복 수가 너무 많아지면 Bottleneck 블록 안에서 주요 정보가 희석될 수 있다.

1. **결론**
   1. **주요 결론**
      1. Bottleneck 반복수 증가에 따른 초기 성능 개선

Bottleneck 반복수가 1, 1, 1, 1에서 1, 2, 2, 1로 증가한 경우는 보존율이 약 10.4% 상승하였다. 이는 NPU가 Bottleneck 구조의 연산에 최적화되어 성능 향상에 기여할 수 있다고 판단한다.

* + 1. Bottleneck 반복수 증가의 한계

Bottleneck 반복수를 1, 2, 2, 1에서 2, 2, 2, 2로 증가한 경우는 보존율이 약 2.4%감소한 것을 볼 수 있다. 이는 Bottleneck 반복수가 증가하며 NPU에서의 연산량이 포화상태에 도달하여 연산 효율성이 저하된 것으로 보인다.

* + 1. Bottleneck 반복수의 과도한 증가의 부정적인 영향

Bottleneck 반복수가 2, 2, 2, 2에서 2, 3, 3, 2로 증가한 경우는 보존율이 약 0.6% 추가 감소한 것이 보인다. 이는 Bottleneck 블록의 과도한 반복이 NPU의 연산에서 연산 부담이 더 커지는 것을 의미한다. 또한 레이어와 레이어 사이에서 주요 정보가 희석되어 성능이 저하될 수 있다.

* 1. **향후 연구 방향**

레이어에서 Bottleneck외의 concat요소에 대한 분석과 다른 다양한 레이어(c2f, c3Ghost)들의 구조를 분석하여 보존율에 어떻게 영향을 끼치는지 알아볼 예정이다. 또한 conv레이어와 Ghostconv레이어를 비교하여 conv를 사용했을 때의 성능과 보존율, Ghostconv를 사용했을 때의 성능을 분석할 예정이다.

1. **참조 문헌**

[1] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," *arXiv preprint arXiv:1712.05877*, 2017.