**Concat 텐서 그룹수와 보존율 분석 보고서**

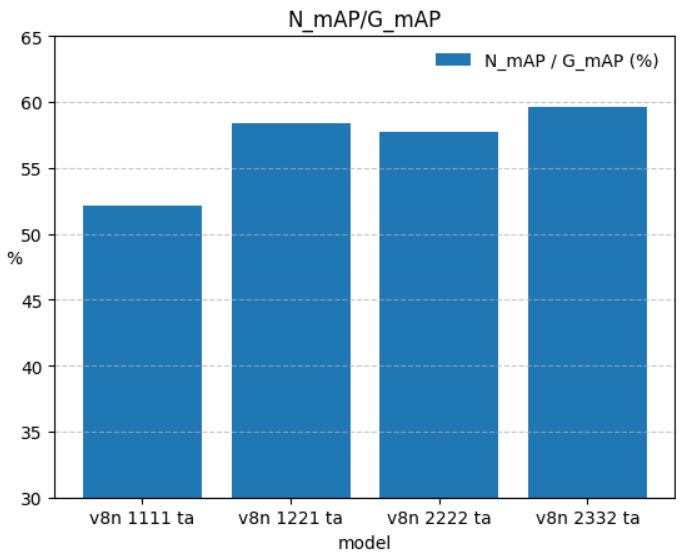
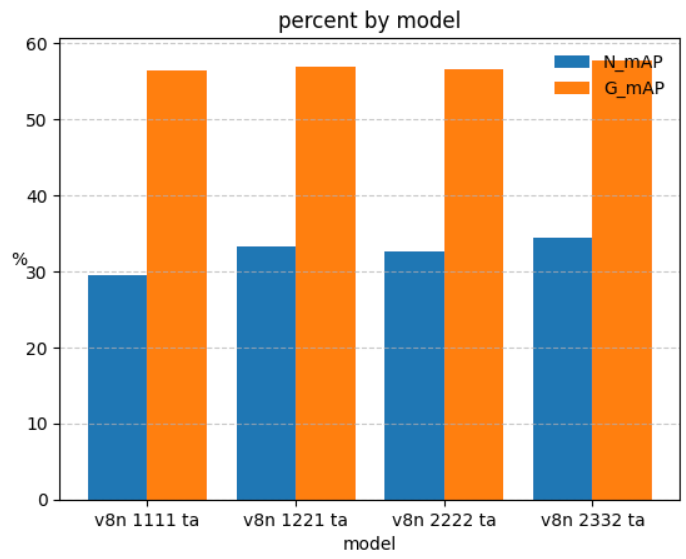
1. **서론**
   1. **배경**
      1. 앞선 보고서에서 c3레이어에 대한 bottleneck 반복수에 따른 보존율 상관관계에 알아보았고, 본 보고서에는 c2f에 대한 내용을 다룰 예정이다. 우선 c2f 레이어의 동작 방식은 cv1을 통해 2개의 하위 텐서로 나뉘는 것은 c3레이어와 동일하지만 c3레이어의 Bottleneck반복을 통한 후 concat하는 것과 달리 텐서 복사본을 생성 후 각각 n개의 복사본이 Bottleneck을 통과한 후 원본 정보를 가지고 있는 텐서와 concat되고 출력된다. 본 보고서는 v8n기준 concat 텐서수에 따른 보존율의 상관관계를 분석할 예정이다.
   2. **실험 준비** 
      1. 실험 모델
         1. v8n\_1111
         2. v8n\_1221
         3. v8n\_2222
         4. v8n\_2332
   3. **목적**

보존율 = NmAP / GmAP

* + 1. concat이 받는 텐서 그룹수와 양자화 손실률 상관관계 분석  
       : concat 텐서 수가 많아질수록 데이터의 범위가 증가 한다. 즉 값의 분포가 넓어질 가능성이 있는데, 이 넓어진 분포를 양자화된 모델이 제대로 표현하지 못 해 정보 손실이 증가할 수 있다. 또한 concat된 텐서 값들이 특정 범위에 치우지거나 분포가 비선형적인 경우에 양자화 과정에서 작은 값들의 정보가 손실되거나 큰 값이 클리핑되어 정보 왜곡이 발생 할 수 있다. 본 실험의 목적은 적절한 concat 텐서수를 알아내기 위해 실험을 진행한다.
  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. concat 텐서수에 따른 연산 보존율 비교
         1. 비교 모델 :
            1. v8n\_1111 vs v8n\_1221 vs v8n\_2222 vs v8n\_2332
         2. 모델 설명 :
            1. v8n\_1111: 기본 yolov8n 모델에 concat 텐서수 1, 1, 1, 1를 설정한 모델
            2. v8n\_1221: 기본 yolov8n 모델
            3. v8n\_2222: 기본 yolov8n 모델에 concat 텐서수 2, 2, 2, 2를 설정한 모델
            4. v8n\_2332: 기본 yolov8n 모델에 concat 텐서수 2, 3, 3, 2를 설정한 모델
         3. **가설** **– concat 텐서수가 증가할수록 보존율이 하락할 것이다.**  
             **concat** 텐서수가 증가하면 모델의 데이터 값들의 분포가 넓어지고, 모델이 학습할 수 있는 정보의 범위가 증가한다. 하지만 8bit 연산 환경에서 범위가 넓어진 데이터 값을 정확히 활용하지 못 할 수 있다. 또한 해당 concat에서의 데이터 값들의 분포가 특정 범위에 치우치거나 outlier 범위에 있을 경우에 작은 값의 정보가 손실되거나 큰 값들이 clipping되어 정보가 왜곡될 수 있다. 따라서 1111모델에서 concat 텐서수가 증가한 모델일수록 보존율 하락폭이 클 것이다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Parameter 수 | NPU/GPU 비율 | NPU 성능 | GPU 성능 | FPS |
| v8n\_1111 | 2.9M | 52.16% | 29.42% | 56.4% | 181.74 |
| v8n\_1221 | 3.01M | 58.37% | 33.2% | 56.88% | 165.44 |
| v8n\_2222 | 3.02M | 57.72% | 32.66% | 56.58% | 162.03 |
| v8n\_2332 | 3.44M | 59.59% | 34.42% | 57.76% | 146.01 |

* + - 1. 결과 :
      2. 분석 :
         1. **v8n\_1111 -> v8n\_1221**

보존율 기준 52.16% 에서 58.37%로 약 6.2% 상승폭을 보인다. Concat 텐서수가 증가함에 따라 NPU에서 활용 가능한 정보가 더 증가한 것으로 보인다. NPU의 병렬 연산이 효율적으로 수행되어 성능 개선에 기여한 것으로 판단된다.

* + - * 1. **v8n\_1221 -> v8n\_2222**

보존율 기준 58.37% 에서 57.72%로 약 0.65% 하락폭을 보인다. Concat 텐서수가 균일하게 증가함에 따라 NPU연산 환경에서 값 분포의 복잡성 증가로 인해 효율이 저하되고 값 분포가 넓어짐에 따라 작은 값의 손실 또는 큰 값의 clipping이 발생했을 가능성이 있다. 또한 GPU에서의 성능도 약간 감소함에 따라 NPU에서의 성능도 저하되어 보존율에도 영향을 미친다. 이는 GPU에서의 연산 효율이 저하된 것으로 판단된다.

* + - * 1. **v8n\_2222 -> v8n\_2332**

보존율 기준 57.72% 에서 59.59%로 약 2% 상승폭을 보인다. Concat 텐서수를 비대칭적으로 조정한 경우에 각 레이어에서 생성되는 데이터의 다양성이 달라져 데이터 값 분포가 더 균형적으로 조정된 것으로 추정한다. 텐서수를 일정하게 늘리는 경우보다 각 레이어의 역할에 따른 텐서수 최적화가 NPU성능 향상에 기여한 것으로 보인다.

1. **결론**
   1. **주요 결론**
      1. 적정 텐서 수 조합

v8n\_2332모델이 가장 높은 보존율 59.59%을 보여준다. 이는 비대칭적으로 텐서수를 조정했을 때 데이터 값 분포가 더 균형적으로 조정되어 정보 손실이 적다고 판단했다. 반대로 균일하게 증가한 v8n\_2222는 하락폭을 보였다. 이는 데이터 값 분포에서 더 다양한 정보를 제공하지만 GPU 연산 효율이 저하되어 보존율이 오히려 감소했다.

* + 1. Concat 텐서 수가 NPU 성능에 미치는 영향

NPU 환경에서의 병렬 연산은 concat 텐서수가 적절하게 조정된 경우 성능을 향상시킨다. 하지만 값 분포의 범위가 지나치게 넓어지거나 작은 값, 혹은 큰 값에 몰려있으면 정보 손실량이 매우 많아저 오히려 연산 효율이 저하되는 상황을 초래한다. 따라서 모델 설계 시 데이터 분포와 NPU 연산 환경을 고려한 concat텐서수를 설정해야 한다.

* 1. **향후 연구 방향**

이번 분석으로 앞선 보고서의 c3레이어의 Bottleneck과 c2f레이어의 concat에 대해 알아보았다. c2f레이어가 c3 레이어에 비해 파라미터 수가 많고 그에 따라 성능도 좋게 나왔다. 이러한 결과를 바탕으로 c2f, c3, c3g레이어에 대한 각 성능과 특징을 분석하고, Ghostconv와 conv 레이어를 비교할 예정이다.