모델 비교 분석 보고서

모델 리스트

|  |  |  |  |
| --- | --- | --- | --- |
| **모델** | **Bottleneck 반복 수** | **Backbone 구조** | **Neck 구조** |
| v8n\_2442 | [2, 4, 4, 2] | C2f | C2f |
| v8n\_4664 | [4, 6, 6, 4] | C2f | C2f |
| v8n\_6886 | [6, 8, 8, 6] | C2f | C2f |
| v8n\_b8n5\_2442 | [2, 4, 4, 2] | C2f | C3 |
| v8n\_b8n5\_4664 | [4, 6, 6, 4] | C2f | C3 |
| v8n\_b8n5\_6886 | [6, 8, 8, 6] | C2f | C3 |
| v5n\_2442 | [2, 4, 4, 2] | C3 | C3 |
| v5n\_4664 | [4, 6, 6, 4] | C3 | C3 |
| v5n\_6886 | [6, 8, 8, 6] | C3 | C3 |
| v5n\_b5n8\_2442 | [2, 4, 4, 2] | C3 | C2f |
| v5n\_b5n8\_4664 | [4, 6, 6, 4] | C3 | C2f |
| v5n\_b5n8\_6886 | [6, 8, 8, 6] | C3 | C2f |

1. 실험 개요
   1. 배경 : 모델의 네트워크 구조를 수정하여 GPU 상에서 성능 결과가 좋아도 NPU 상에서의 성능이 꼭 좋다고 할 수 없다. 일반적으로 GPU는 부동소수점 연산에 최적화되어 있어 8bit 양자화 연산 시에도 성능 저하가 크지 않다. 하지만 NPU의 경우 연산 능력 및 구조가 GPU와 다르기 때문에 8bit 양자화를 적용하였을 때 특정 모델 구조에서는 성능 저하가 크게 나타날 수 있다. 따라서 YOLO 모델의 다양한 구조가 GPU 대비 NPU에서 얼마나 성능 차이를 유발하는지 분석하여 NPU에 적합한 구조를 찾기 위해 실험할 필요가 있다.
   2. 목적 : 모델의 다양한 구조들이 각각 GPU 대비 NPU에서 성능 차이에 어떤 영향을 미치는지 분석하고, 특정 구조 및 설정이 오차를 유발하는 요인을 찾는다. 특히 Bottleneck 반복 수와 Backbone & Neck에서의 여러 실험을 통해 Bottleneck과 C2f, C3가 성능 차이에 영향을 얼마나 미치는지 확인한다. 이를 통해 GPU에서는 성능이 우수하나 NPU에서는 하락하는 구조를 식별한다. 이로써 8bit 연산에 적합하거나 그렇지 않은 구조를 구분하고, NPU 성능의 최적하에 기여할 수 있는 구조적 특징을 도출한다.
2. 실험 세팅?
   1. 모델 구조 : 비교하는 모델의 세부적인 구조 설명 (예: v8n\_2442 vs v8n\_4664, v5n\_2442 vs v5n\_4664 등).
   2. 양자화 및 구현 방법 : 모델을 8비트로 양자화하여 실험하는 과정과 각 모델에서 사용된 기법을 기술합니다.
3. 실험 기준
   1. 비교 기준 :
   2. 설명 : 각 기준이 성능에 미치는 영향을 분석하고, 이를 통해 NPU와 GPU에서 어떤 구조가 성능 차이를 일으키는지 파악한다. 실험의 목적은 GPU에서 우수한 성능을 보이지만 NPU에서는 성능 저하가 큰 구조를 찾아내어, 해당 구조가 8비트 연산에 적합하지 않다는 결론을 도출하는 것이다.
4. 실험 결과

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Npu/gpu | Npu map | Gpu map |
| V8n\_b5n8\_2442 | 0.554335895 | 0.303 | 0.5466 |
| V8n\_b5n8\_4664 | 0.53974665 | 0.294 | 0.5447 |
| v5n\_b8n5\_4664 | 0.579024035 | 0.318 | 0.5509 |
| v5n\_b8n5\_2442 | 0.55545471 | 0.306 | 0.5473 |
| v5n\_4664 | 0.604196816 | 0.334 | 0.5528 |
| V8n\_4664 | 0.599498926 | 0.335 | 0.5588 |
| V8n\_2442 | 0.53162732 | 0.295 | 0.5549 |
| v5n\_2442 | 0.544481358 | 0.295 | 0.5418 |

1. **Bottleneck 수 비교 (2442 vs 4664)**

* **v5n\_2442 vs v5n\_4664**
  + NPU/GPU 차이: 5.97%
  + NPU mAP 차이: 3.9%
  + GPU mAP 차이: 1.1%
* **v8n\_2442 vs v8n\_4664**
  + NPU/GPU 차이: 6.79%
  + NPU mAP 차이: 4%
  + GPU mAP 차이: 0.39%
* **v5n\_b8n5\_2442 vs v5n\_b8n5\_4664**
  + NPU/GPU 차이: 2.4%
  + NPU mAP 차이: 1.2%
  + GPU mAP 차이: 0.36%
* **v8n\_b5n8\_2442 vs v8n\_b5n8\_4664**
  + NPU/GPU 차이: -1.46%
  + NPU mAP 차이: -0.9%
  + GPU mAP 차이: -0.1%

**2) C3, C2f 비교**

* **v5n\_2442 vs v8n\_2442**
  + NPU/GPU 차이: 1.29%
  + NPU mAP 차이: 0%
  + GPU mAP 차이: -1.31%
* **v5n\_4664 vs v8n\_4664**
  + NPU/GPU 차이: 0.6%
  + NPU mAP 차이: -0.1%
  + GPU mAP 차이: -0.6%

**3) C3, C2f 와 Bottleneck 조합 비교**

* **C2f + 4664 vs C3 + 2442 (v5n\_2442 vs v8n\_4664)**
  + NPU/GPU 차이: 5.5%
  + NPU mAP 차이: 4%
  + GPU mAP 차이: 1.7%
* **C3 + 4664 vs C2f + 2442 (v5n\_4664 vs v8n\_2442)**
  + NPU/GPU 차이: 7.26%
  + NPU mAP 차이: 3.9%
  + GPU mAP 차이: -0.21%

**4) Backbone & Neck 조합 비교 (bC2f + nC3 vs bC3 + nC2f)**

* **bC2f + nC3 vs bC3 + nC2f (v5n\_b8n5\_2442 vs v8n\_b5n8\_2442)**
  + NPU/GPU 차이: 0.12%
  + NPU mAP 차이: 0.3%
  + GPU mAP 차이: 0.07%
* **bC2f + nC3 vs bC3 + nC2f (v5n\_b8n5\_4664 vs v8n\_b5n8\_4664)**
  + NPU/GPU 차이: 3.93%
  + NPU mAP 차이: 2.4%
  + GPU mAP 차이: 0.62%
* **(bC2f + nC3) + 2442 vs (bC3 + nC2f) + 4664 (v5n\_b8n5\_2442 vs v8n\_b5n8\_4664)**
  + NPU/GPU 차이: 1.58%
  + NPU mAP 차이: 1.2%
  + GPU mAP 차이: 0.26%
* **(bC2f + nC3) + 4664 vs (bC3 + nC2f) + 2442 (v5n\_b8n5\_4664 vs v8n\_b5n8\_2442)**
  + NPU/GPU 차이: 2.47%
  + NPU mAP 차이: 1.5%
  + GPU mAP 차이: 0.43%

**5) 기타 비교 (Bottleneck과 Neck 구조 추가)**

* **v8n\_b5n8\_2442 vs v8n\_2442**
  + NPU/GPU 차이: 2.27%
  + NPU mAP 차이: 0.8%
  + GPU mAP 차이: -0.83%
* **v8n\_b5n8\_4664 vs v8n\_4664**
  + NPU/GPU 차이: -5.98%
  + NPU mAP 차이: -4.1%
  + GPU mAP 차이: -1.41%
* **v8n\_b5n8\_2442 vs v8n\_4664**
  + NPU/GPU 차이: -4.52%
  + NPU mAP 차이: -3.2%
  + GPU mAP 차이: -1.22%
* **v8n\_b5n8\_4664 vs v8n\_2442**
  + NPU/GPU 차이: 0.81%
  + NPU mAP 차이: -0.1%
  + GPU mAP 차이: -1.02%
* **v5n\_b8n5\_2442 vs v5n\_2442**
  + NPU/GPU 차이: 1.1%
  + NPU mAP 차이: 1.1%
  + GPU mAP 차이: 0.55%
* **v5n\_b8n5\_4664 vs v5n\_4664**
  + NPU/GPU 차이: -2.52%
  + NPU mAP 차이: -1.6%
  + GPU mAP 차이: -0.19%
* **v5n\_b8n5\_2442 vs v5n\_4664**
  + NPU/GPU 차이: -4.87%
  + NPU mAP 차이: -2.8%
  + GPU mAP 차이: -0.55%
* **v5n\_b8n5\_4664 vs v5n\_2442**
  + NPU/GPU 차이: 3.45%
  + NPU mAP 차이: 2.3%
  + GPU mAP 차이: 0.91%

1. 결과 분석

1) Bottleneck 수 비교 (2442 vs 4664)

* v5n 및 v8n 모델
  + Bottleneck 수가 더 많은 4664 모델이 대부분의 경우에서 더 높은 NPU/GPU 성능 비율을 보인다.
  + 이는 Bottleneck 수가 증가함에 따라 NPU의 8비트 연산 효율이 더 높아진다는 것을 암시한다.
* v8n\_b5n8 모델
  + Bottleneck 수가 4664일 때 모델이 오히려 NPU/GPU 성능 비율이 감소(-1.46%)하였다.

요약: Bottleneck 수가 많은 모델(4664)이 대부분 8bit 연산에서 높은 NPU/GPU 성능 비율을 보인다. 하지만, v8n\_b5n8 모델의 경우 오히려 낮게 나왔다.

2) C3 vs C2f 비교

* v5n 및 v8n 모델
  + Bottleneck 반복 수가 4664, 2442일 때 v5n 모델이 v8n 모델 보다 높은 NPU/GPU 성능 나왔다. 하지만 각각의 성능 차이가 크지 않으므로 유의미한 정보는 아닌 것을 판단된다.

요약: Bottleneck 반복 수가 4664, 2442일 때 v5n과 v8n의 차이가 크지 않았다.

3). C3, C2f 와 Bottleneck 조합 비교 (C3 + 2442 vs C2f + 4664 / C3 + 4664 vs C2f + 2442)

* C2f + 4664 vs C3 + 2442 (v8n\_4664 vs v5n\_2442)
  + C2f + 4664(v8n\_4664) 모델이 C3 + 2442(v5n\_2442) 모델보다 NPU/GPU 성능 비율이 5.5% 높다.
  + NPU mAP는 4% 향상되었다.
* C3 + 4664 vs C2f + 2442 (v5n\_4664 vs v8n\_2442)
  + C3 + 4664(v5n\_4664) 모델이 C2f + 2442(v8n\_2442) 보다 NPU/GPU 성능 비율이 7.26%로 큰 증가를 보인다.
  + GPU mAP는 0.21% 감소하였다

.

분석 요약: C3 + 4664 조합이 8비트 연산에서 가장 높은 NPU/GPU 성능 비율을 보인다.

4) Backbone & Neck 조합 비교 (bC2f + nC3 vs bC3 + nC2f)

* bC2f + nC3 vs bC3 + nC2f (v5n\_b8n5\_2442 vs v8n\_b5n8\_2442)
  + bC2f + nC3 조합이 bC3 + nC2f 조합보다 성능이 높게 나오지만 차이가 매우 미미하다.
  + Bottleneck 반복 수가 2442일 때 두 구조의 연산 효율에 큰 차이가 없는 것으로 판단된다.
* bC2f + nC3 vs bC3 + nC2f (v5n\_b8n5\_4664 vs v8n\_b5n8\_4664)
  + Bottleneck의 반복 수가 4664로 증가한 경우, (bC2f + nC3) 조합이 (bC3 + nC2f) 조합에 비해 NPU/GPU 성능 비율, NPU mAP, GPU mAP 이 더 높다. NPU/GPU 성능 비율이 3.93% 증가하였으며 이는, 4664의 Bottleneck 수가 NPU의 병렬 연산 효율을 높이는데 유리하게 작용했음을 볼 수 있다.
* (bC2f + nC3) + 2442 vs (bC3 + nC2f) + 4664 (v5n\_b8n5\_2442 vs v8n\_b5n8\_4664)
  + v5n\_b8n5\_2442가 v8n\_b5n8\_4664 보다 NPU/GPU 성능 비율이 1.58% 증가하였다. NPU, GPU mAP는 각각 1.2%, 0.26% 증가하였다.
  + (bC2f + nC3) + 2442 조합이 Bottleneck 수가 더 적음에도 불구하고 (bC3 + nC2f) + 4664 조합보다 모든 측면에서 약간 더 나은 성능을 보인다. bC2f + nC3 조합이 단순히 Bottleneck 수의 증가에 의존하지 않고도 높은 효율을 발휘하는 것으로 보아 8bit 연산 효율 측면에서 Bottleneck의 증가가 필요치 않을 때 유리한 구조인 것으로 판단된다.
* (bC2f + nC3) + 4664 vs (bC3 + nC2f) + 2442 (v5n\_b8n5\_4664 vs v8n\_b5n8\_2442)
  + v5n\_b8n5\_4664이 vs8n\_b5n8\_2442보다 NPU/GPU 성능 비율이 2.47% 증가하였고, NPU, GPU의 mAP값은 각각 1.5%, 0.43% 증가하였다.
  + 위의 결과에서 나아가 Bottleneck 수가 더 많을 때에도 bC2f + nC3 8비트 연산에서의 성능이 더 높은 것으로 보아, bC2f + nC3 조합이 8bit 연산에 적합한 것으로 판단된다.

5) 기타 비교 (Bottleneck과 Neck 구조 추가)

* v8n\_b5n8\_2442 vs v8n\_2442
  + v8n\_b5n8\_2442 모델이 v8n\_2442 모델보다 NPU/GPU 성능 비율이 2.27% 높다.
  + 같은 Bottleneck 반복 수 2442에서 bC3 + nC2f 구조가 bC2f + nC2f 구조보다 NPU/GPU 성능 비율이 높은 것을 보아 8bit 연산에 더 적합한 것으로 판단된다.
* v8n\_b5n8\_4664 vs v8n\_4664
  + 같은 Bottleneck 반복 수 4664에서는 오히려 bC2f + nC2f 구조가 bC3 + nC2f구조보다 NPU/GPU 성능 비율이 5.98% 높다. 또한 NPU, GPU mAP 성능은 4.1%, 1.41% 증가하였다.
  + 오히려 Bottleneck수가 4664로 증가하였을 때, 위의 결과와 반대로 bC2f+ nC2f 구조가 bC3 + nC2f 구조보다 8bit 연산에서 성능이 더 좋았다.
  + bC2f + nC2f vs bC3 + nC2f 조합에서는 bottleneck 반복 수에 따라 결과가 반대로 나온다.
* v8n\_b5n8\_2442 vs v8n\_4664
  + bC2f + nC2f + 4664 조합이 bC3 + nC2f + 2442 조합보다 NPU/GPU 성능 비율이 4.52% 높다. NPU, GPU mAP는 각각 3.2%, 1.22% 높다.
  + 같은 nC2f를 가진 구조일 때, Backbone이 C2f이고 Bottleneck 반복 수가 높은 것이 8bit 연산에 효율적인 것으로 판단된다.
* v8n\_b5n8\_4664 vs v8n\_2442
  + bC3 + nC2f + 4664 조합이bC2f + nC2f + 2442 조합보다 NPU/GPU 성능 비율이 0.81% 높다. NPU, GPU mAP에서는 오히려 성능이 감소하였지만 세 가지 값 모두 작은 값이기 때문에 성능에 영향이 미미한 것으로 판단된다.
  + 작은 값이지만 nC2f를 가진 구조에서 Backbone을 C3을 사용하고 Bottleneck 반복 수가 많은 조합이 8bit 연산 효율에 더 좋다.
* v5n\_b8n5\_2442 vs v5n\_2442
  + bC2f + nC3 + 2442 조합이 bC3 + nC3 + 2442 조합보다 NPU/GPU 성능비율이 1.1% 높다. NPU, GPU에서는 각각 1.1%, 0.55% 더 좋다.
  + 마찬가지로 미미한 차이이지만 같은 bottleneck에서 nC3구조에서 Backbone에서 C2f를 사용하는 것이 8bit 측면에서 더 좋다.
* v5n\_b8n5\_4664 vs v5n\_4664
  + bC3 + nC3 + 4664 조합이 bC3 + nC2f + 4664 조합에서NPU/GPU 성능 비율이 2.52% 높다. NPU, GPU에서는 각각 1.6%, 0.19% 더 좋다.
  + 같은 bC3 구조에서 같은 4664 반복 수를 사용하면 nC3를 사용하는 구조가 8bit 측면에서 성능이 더 좋다.
* v5n\_b8n5\_2442 vs v5n\_4664
  + bC3 + nC3 + 4664 조합이 bC2f + nC3 + 2442 조합보다 NPU/GPU 성능 비율이 4.87% 높다. NPU, GPU 성능은 각각 2.8%, 0.55% 좋다.
  + 같은 nC3 구조에서 Bottleneck반복 수가 더 많고Backbone에서 C3을 사용하는 구조가 더 8bit에서 성능이 좋다.
* v5n\_b8n5\_4664 vs v5n\_2442
  + bC2f + nC3 + 4664 조합이 bC3 + nC3 + 2442 조합보다 NPU/GPU 성능 비율이 3.45% 높다. NPU, GPU는 각각 2.3%, 0.91% 높다.
  + 같은 nC3 구조에서 Botteleneck 구조에서 더 많은 반복 수를 사용하고 Backbone에서 C2f가 사용된 것이 8bit 연산에 효율적이다.

1. 분석 및 결론

위의 과정을 통해 다양한 Backbone 및 Neck 구조와 Bottleneck 반복 수가 NPU와 GPU의 8비트 연산 성능에 미치는 영향을 평가하였다. 분석 및 결론에서는 각 조합을 비교하며 Bottleneck 반복 수, Backbone 및 Neck 조합이 성능에 미치는 영향을 항목별로 자세히 평가하였으며, 그에 따라 8비트 연산에서 최적화된 구조를 도출하였다.

1) Bottleneck 반복 수의 영향: 2442 vs 4664

비교 항목:

* Bottleneck 수가 2442일 때와 4664일 때 성능을 비교하였다.
* 비교 대상 모델은 주로 v5n과 v8n 모델이었으며, 단순히 Bottleneck 수만 늘렸을 때의 성능 차이를 확인했다.

결과 및 분석:

* Bottleneck 수가 많은 4664 모델이 대체로 더 높은 NPU/GPU 성능 비율을 보였으며, 이는 8bit 연산에서 효율을 높이는 데 Bottleneck 반복 수가 많은 것이 성능에 좋다고 판단할 수 있다.
* 하지만, v8n\_b5n8 모델의 경우 Bottleneck 수가 4664일 때 오히려 성능 비율이 감소(-1.46%)하는 현상이 관찰되었다. 이는 특정 조합에서는 Bottleneck 반복 수가 많은 것이 오히려 NPU의 병렬 처리 효율을 저하시킬 수 있음을 의미한다.

결론:

* 대체로 Bottleneck 반복 수가 많을수록 NPU 성능에 긍정적인 영향을 끼치지만, 특정 구조에서는 성능 저하를 방지하기 위해 적절한 Bottleneck 반복 수 조정이 필요하다.

2) v5n, v8n 구조 비교: C3 vs C2f

비교 항목:

* 동일한 Bottleneck 수(2442 및 4664) 조건에서 C3와 C2f을 사용한 모델들을 비교하여, v5n과 v8n의 성능 차이를 분석하였다.

결과 및 분석:

* v5n과 v8n 모델에서 Bottleneck 반복 수가 2442와 4664일 때 모두 성능 차이가 거의 없었다.
* 이는 C3, C2f의 차이만 봤을 때, 성능에 큰 차이가 없는 것으로 판단된다.

결론:

* Backbone 구조만의 변경(C3 vs C2f)은 8비트 연산 최적화에 유의미한 성능 차이를 제공하지 않으며, 성능 최적화를 위해서는 Bottleneck과 Neck 조합을 신중히 설계해야 한다.

3) Backbone과 Bottleneck 조합 비교: C3 + 2442 vs C2f + 4664 / C3 + 4664 vs C2f + 2442

비교 항목:

* C3와 C2f Backbone 구조에 각각 2442와 4664의 Bottleneck 수를 조합하여 성능을 비교하였다. 이 조합이 NPU/GPU 성능 비율에 미치는 영향을 확인하였다.

결과 및 분석:

* \*\*C2f + 4664 조합(v8n\_4664)\*\*이 \*\*C3 + 2442 조합(v5n\_2442)\*\*에 비해 NPU/GPU 성능 비율에서 5.5% 더 높은 성능을 보였다.
* \*\*C3 + 4664 조합(v5n\_4664)\*\*은 \*\*C2f + 2442(v8n\_2442)\*\*보다 7.26% 높은 NPU/GPU 성능 비율을 기록하였다.
* 이는 특정 Bottleneck 반복 수와 Backbone 조합이 결합될 때 NPU의 병렬 처리 효율을 극대화할 수 있음을 나타낸다.

결론:

* Bottleneck 반복 수가 많은 경우(4664), C3 구조가 NPU의 8비트 연산에서 최적화된 성능을 보이며, C3 + 4664 조합이 가장 효과적인 조합으로 도출되었다.

4) Backbone과 Neck 조합 비교: bC2f + nC3 vs bC3 + nC2f

비교 항목:

* Backbone이 bC2f일 때와 bC3일 때 각각 Neck 조합을 nC3와 nC2f로 비교하였다. Bottleneck 수가 2442와 4664일 때 성능을 비교하여, 조합별 성능의 차이를 분석하였다.

결과 및 분석:

* Bottleneck 반복 수가 2442와 4664일 때 bC2f + nC3 조합이 bC3 + nC2f 조합에 비해 더 높은 NPU/GPU 성능 비율을 보였다.
* 특히 Bottleneck이 4664일 때, bC2f + nC3 조합은 NPU/GPU 성능 비율을 3.93% 향상시켰으며, NPU 및 GPU mAP 성능에서도 더 높은 결과를 나타냈다.
* 또한, (bC2f + nC3) + 2442 조합이 (bC3 + nC2f) + 4664 조합보다 높은 성능을 보여, Bottleneck 수가 적더라도 복잡한 조합을 사용하지 않고도 안정적인 성능을 발휘할 수 있음을 시사하였다.

결론:

* Backbone이 bC2f이고 Neck이 nC3인 조합이 특히 Bottleneck 수가 많은 경우에 8비트 연산 최적화에 가장 적합한 구조이다.

5) 복잡한 Bottleneck과 Neck 구조 추가의 부정적 영향

비교 항목:

* 복잡한 Bottleneck과 Neck 구조를 추가한 v8n\_b5n8 및 v5n\_b8n5 모델들과 단순한 구조 모델들을 비교하였다. 구조의 복잡성이 성능에 미치는 영향을 분석하였다.

결과 및 분석:

* 복잡한 구조가 추가된 v8n\_b5n8 및 v5n\_b8n5 모델에서는 대부분 NPU/GPU 성능 비율이 감소하는 결과를 보였다.
* v8n\_b5n8\_4664 모델의 경우, 복잡한 구조로 인해 NPU/GPU 성능 비율이 5.98% 감소하였으며, 이는 구조적 복잡성이 NPU의 병렬 처리 효율에 부정적인 영향을 미친다는 것을 보여준다.

결론:

* 복잡한 Bottleneck과 Neck 구조는 NPU의 8비트 연산에서 성능을 저하시키며, 특히 단순한 구조일 때 더 높은 성능을 발휘할 가능성이 크다.

최종 결론

8비트 연산에 최적화된 NPU 성능을 달성하기 위해서는 Bottleneck 반복 수, Backbone 및 Neck 조합의 균형이 매우 중요하다. 각 항목을 통해 도출한 결론을 종합하여 다음과 같은 최적화 경향을 확인하였다.

1. Bottleneck 반복 수: 더 많은 반복 수(4664)가 대체로 8비트 연산에서 성능을 향상시키는 데 유리하다. 그러나 복잡한 구조에서는 성능 저하가 발생할 수 있으므로, 구조에 따라 반복 수를 조절하는 것이 필요하다.
2. Backbone 및 Neck 조합: bC2f + nC3 조합이 특히 8비트 연산에서 효율이 높으며, Bottleneck 수가 많을 때도 성능 저하 없이 NPU 성능을 극대화할 수 있다. 이는 병렬 연산 최적화를 위해 bC2f + nC3 조합이 가장 적합한 선택임을 시사한다.
3. 복잡한 구조의 부정적 영향: 복잡한 Bottleneck과 Neck 구조는 오히려 성능을 저하시킬 수 있다. 특히 NPU의 효율이 중요한 8비트 연산에서는 복잡한 구조보다 단순한 구조가 적합하다.

최종 권장 사항: 8비트 연산에서 최적의 성능을 위해 Bottleneck 반복 수가 4664인 단순한 구조와 bC2f + nC3 조합을 사용하는 것이 가장 효율적이며, 과도한 구조적 복잡성은 피하는 것이 바람직하다.

* 1. 향후 방향 : 추가 실험이나 분석이 필요한 요소와 향후 연구 방향을 설명합니다.

1. 참고 문헌