학습 돌린 모델 list

* V5n\_c2f vs v8n (p2 x)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model\_name | Back | neck | Npu/gpu | npu | Gpu |
| V8n | C2f | C2f |  |  |  |
| V5n\_c2f | C2f | C2f |  |  |  |

* V8n\_c3 vs v5n (p2 x)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model\_name | back | neck | Npu/gpu | npu | Gpu |
| V5n | C3 | C3 |  |  |  |
| V8n\_c3 | C3 | c3 |  |  |  |

가설 : 넥에 있는 conv2 두 레이어가 연산 양자화 손실에 영향이 없을 것이다.

변인통제 : 기법은 모두 같고 첫 input tensor 크기와, 넥의 conv 두 레이어만 다름.

확인사항 :

학습 돌린 모델 list

* 8SP2  
  train vs (train + val)

가설 : 데이터셋 영향력

변인통제 : train set만

확인사항 : bbox별 정확도 + 전체 정확도

학습 돌린 모델 list (이 list의 4가지 모델들 모두 width=0.4 주고 학습한 모델)

* v8s\_c3\_conv vs v8s\_c3g\_conv (p2 o)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model\_name | Npu/gpu | npu | Gpu | params | fps |
| V8s\_c3\_conv | 0.623288774 | 0.387 | 0.6209 | 5.666 | 74.63 |
| V8s\_c3g\_conv | 0.613346418 | 0.375 | 0.6114 | 5.0006 | 77.99 |

가설 : c3ghost가 c3 보다 연산 양자화 손실률이 적을 것이다. 연산량이 적기 때문.

변인통제 : backbone, neck 전체 c3 vs c3g

확인사항 : gpu, npu, 보존율(+ params, fps)

결과 : c3\_conv – c3g\_conv

* + gpu : 0.95%
  + npu : 1.2%
  + 보존율 : 1%
  + Params : 0.6M
  + FPS : -3.36

결론 : conv가 같은 조건에서, c3가 c3g보다 fps를 제외하고 미미한 차이로 좋다. FPS를 고려하면 c3g\_conv가 real time에서 더 좋은 것으로 판단된다.

* v8s\_c3\_gconv vs v8s\_c3g\_gconv (p2 o)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model\_name | Npu/gpu | npu | Gpu | params | FPS |
| V8s\_c3\_gconv | 0.476734694 | 0.292 | 0.6125 | 5.1547 | 74.27 |
| V8s\_c3g\_gconv | 0.551386623 | 0.338 | 0.613 | 4.4894 | 74.86 |

가설 : c3ghost가 c3 보다 연산 양자화 손실률이 적을 것이다. 연산량이 적기 때문.

변인통제 : Backbone, Neck 전체 c3 vs c3g

확인사항 : gpu, npu, 보존율(+ params, FPS)

결과 : c3g\_gconv – c3\_gconv

* + gpu : 0.05%
  + npu : 4.6%
  + 보존율 : 3.67%
  + Params : - 0.66M
  + FPS : 0.59

결론 : gconv가 같은 조건에서, c3g가 c3보다 더 좋다. GPU 성능 차이는 없지만 나머지 성능에서 모두 c3g가 높다.

* v8s\_c3\_conv vs v8s\_c3\_gconv (p2 o)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model\_name | Npu/gpu | npu | Gpu | params | FPS |
| V8s\_c3\_conv | 0.623288774 | 0.387 | 0.6209 | 5.666 | 74.63 |
| V8s\_c3\_gconv | 0.476734694 | 0.292 | 0.6125 | 5.1547 | 74.27 |

가설 : conv가 ghostconv보다 연산 양자화 손실이 많을 것이다. 연산량이 많기 때문.

변인통제 : 백 넥 전체 conv vs gconv

확인사항 : gpu, npu, 보존율(+ params, FPS)

결과 : c3\_conv – c3\_gconv

* + gpu : 0.84%
  + npu : 9.5%
  + 보존율 : 14.66%
  + Params : 0.513M
  + FPS : 0.36

결론 : c3가 같은 조건에서, conv의 성능이 ghostconv보다 월등히 높다. Conv가 Gconv보다 연산 양자화 손실이 적다. Ghostconv는 concat 하는 연산이 있기 때문에 손실이 일어나는 것으로 예상.

* v8s\_c3g\_conv vs v8s\_c3g\_gconv (p2 o)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model\_name | Npu/gpu | npu | Gpu | params | FPS |
| V8s\_c3g\_conv | 0.613346418 | 0.375 | 0.6114 | 5.0006 | 77.99 |
| V8s\_c3g\_gconv | 0.551386623 | 0.338 | 0.613 | 4.4894 | 74.86 |

가설 : 위의 결론에 따라 conv가 ghostconv 연산 양자화 손실이 적을 것이다.

확인사항 : gpu, npu, 보존율(+ params, FPS)

결과 : c3g\_conv – c3g\_gconv

* + gpu : -0.16%
  + npu : 3.7%
  + 보존율 : 6.19%
  + Params : 0.5112M
  + FPS : 3.13

결론 : c3g가 같은 조건에서 conv가 ghostconv보다 성능이 좋다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model\_name | Params | Npu/gpu | npu | Gpu | FPS |
| V8s\_c3\_conv | 5.666 | 0.623288774 | 0.387 | 0.6209 | 74.63 |
| V8s\_c3\_gconv | 5.1547 | 0.476734694 | 0.292 | 0.6125 | 74.27 |
| V8s\_c3g\_conv | 5.0006 | 0.613346418 | 0.375 | 0.6114 | 77.99 |
| V8s\_c3g\_gconv | 4.4894 | 0.551386623 | 0.338 | 0.613 | 74.86 |

최종 결론 : gconv는 cv1, cv2 두 단계로 나누고 concat 하는 구조이므로 연산 양자화 손실이 일어난다.

지금 c3\_gconv가 가장 양자화 손실이 크다. 파라미터수는 2번째로 많은데, 보존율 최하위, fps도 최하위.

v8s\_c3g\_conv는 가장 높은 FPS(77.99), 상대적으로 적은 파라미터 수, 준수한 보존율을 유지하므로 위의 4가지 모델 중 가장 성능이 준수한 것으로 판단된다.

1등 c3g\_conv  
2등 c3\_conv  
3등 c3g\_gconv  
4등 c3\_gconv

학습 돌려야 할 모델 list

* 8sp2(train+valid) => t+v 이 좋아야 함.

가설 :

변인통제 : 백 넥 전체 c3 vs c3g

확인사항 : gpu, npu, 보존율(+ params)