Codeitteam7 [AI] 헬스잇(Health Eat) 알약 검출 프로젝트: 내부 실험 보고서

팀장: 이승완(EDA, 옵티마이저)

팀원: 박병호(EDA) 오현민(이미지 크기, 학습률, mosaic 증강) 이경식(yolo 모델 cost) 최준영(mosaic 제외 모든 증강, 최종 검증)

프로젝트 신규 목표: Kaggle mAP 점수를 넘어, '모델 안정성(Lowest Validation Loss)'과 '학습 효율성(Minimum Cost)'을 확보하는 최적의 아키텍처 및 하이퍼파라미터를 탐색한다.

1. 프로젝트 배경: *mAP 0.99의 함정*

1.1 초기 성과

본 프로젝트는 `YOLOv8s` 기반 2-Stage 모델을 활용한 초기 실험에서 Kaggle 리더보드 점수 **0.98613**을 기록하며 높은 성능을 확인하였다.

1.2 문제 발견

뒤에서 알려진 사실이 **Test 셋과 Train 셋의 데이터 유사도가 매우 높다**는 사실이 확인되었다. 이로 인해 모델은 실제 일반화 능력을 보여주기보다는 **Train 셋을 암기(Overfitting)** 하는 방식으로 높은 점수를 달성한 것으로 판단된다. 로컬 환경에서도 mAP 0.99에 어떤 식으로든 쉽게 도달하는 현상이 반복적으로 관찰되었다.

1.3 목표 전환

이에 따라 본 프로젝트는 Kaggle 데이터셋 기반 경쟁을 중단하고, 자체적으로 구축한 `train / val / test` 데이터셋(1472개를 70:15:15 stratified 분할)을 활용하여 내부 실험을 진행한다.

새로운 목표는 다음과 같다.

- **모델 안정성 확보:** Validation Loss 최소화
- **학습 효율성 강화:** 동일 성능 대비 최소 비용 달성
- **높은 mAP**

2. 핵심 EDA 발견: mAP 0.99의 함정 (EDA 담당: 박병호, 이승완)

초기 EDA 과정에서 데이터셋의 구조적 한계가 확인되었다. 분석 결과, 데이터에는 **'3대 편향(Bias)'과 신뢰도 문제**가 존재했으며, 특히 이 3대 편향은 모델이 실제 일반화 능력을 갖추지 못했음에도 불구하고 **mAP 0.99라는 비현실적 성능**을 기록하게 만든 핵심 요인으로 작용했다.

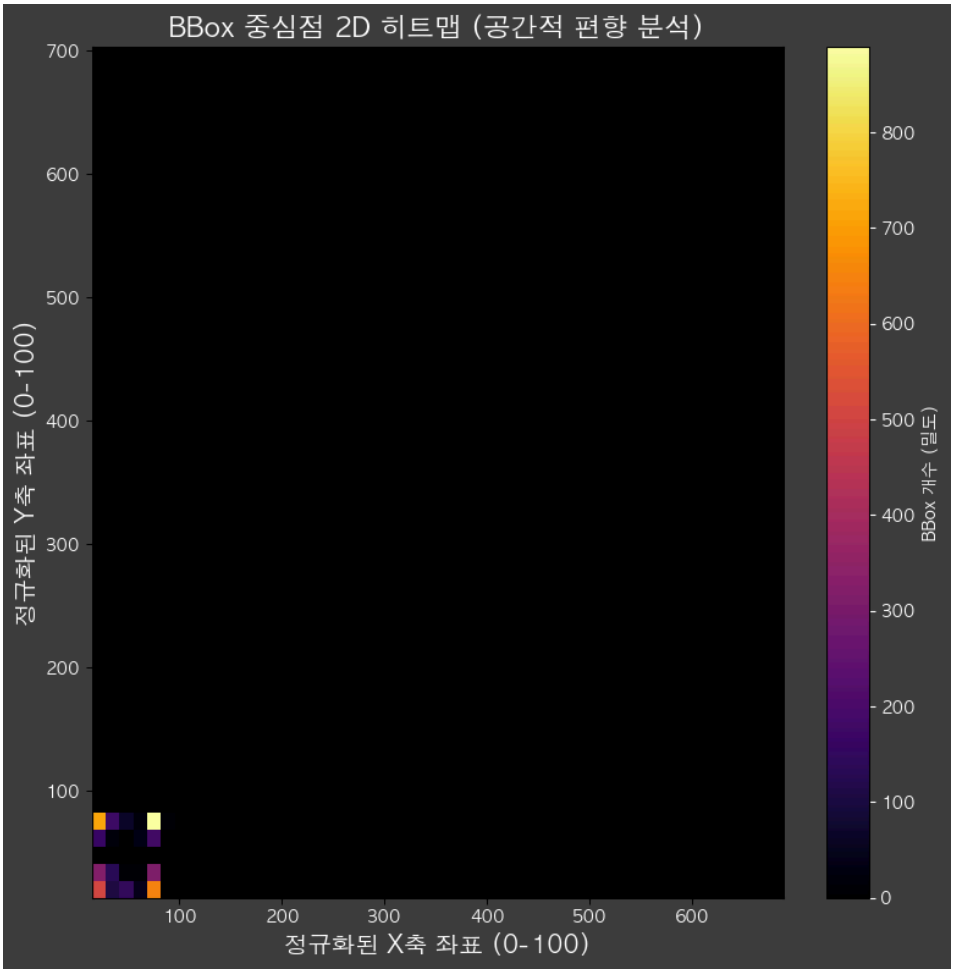
2-1. 발견 1: 극단적인 '환경' 편향

```
1. 캡처 환경 속성별 고유값 개수 (다양성)
- back_color: 1개의 고유값
- light_color: 1개의 고유값
- camera_la : 3개의 고유값
- camera_lo : 1개의 고유값
- drug_dir  : 2개의 고유값
- size      : 1개의 고유값

2. 배경색 (back_color) 분포 (Top 5)
back_color
연회색 배경    1489

3. 조명색 (light_color) 분포
light_color
주백색    1489

4. 카메라 각도 및 크기 (camera_la, camera_lo, size) 통계
         camera_la  camera_lo    size
count  1489.000000     1489.0  1489.0
mean     78.331095        0.0   200.0
std       8.498805        0.0     0.0
min      70.000000        0.0   200.0
25%      70.000000        0.0   200.0
50%      75.000000        0.0   200.0
75%      90.000000        0.0   200.0
max      90.000000        0.0   200.0
```

모든 `train_images` (1,489개)는 완벽하게 통제된, **비현실적인 "실험실" 환경**에서 촬영되었다.
- **이미지 크기:** 모든 이미지가 `976x1280`로 100% 동일
- **배경:** 모든 이미지가 `'연회색 배경'` (1개 고유값)
- **조명:** 모든 이미지가 `'주백색'` (1개 고유값)
- **카메라 각도(lo):** `0` (1개 고유값)

결론: 모델이 배경, 조명 등 '노이즈(Noise)'를 구별할 필요가 거의 없다. 오직 '알약 자체의 특징'에만 집중하면 되는, 비현실적으로 정제된 환경인 것이다. 이렇게 되면 암기만 하기가 한층 더 쉬워진다. 즉 과적합이 나기가 더 쉬워진다.

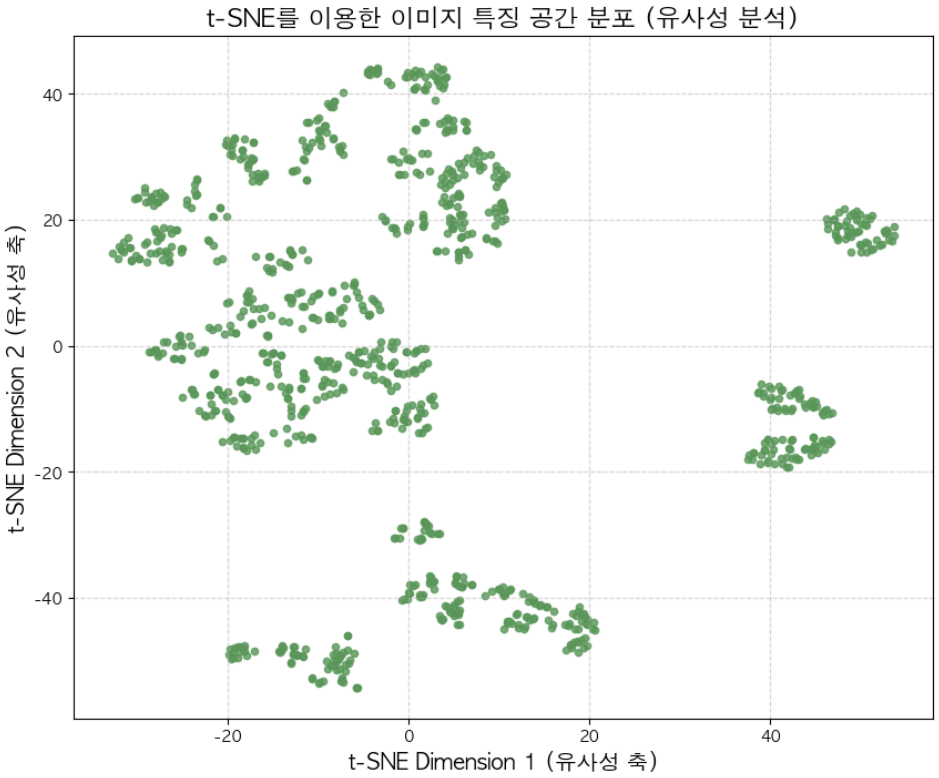
2-2. 발견 2: 극단적인 '공간' 편향



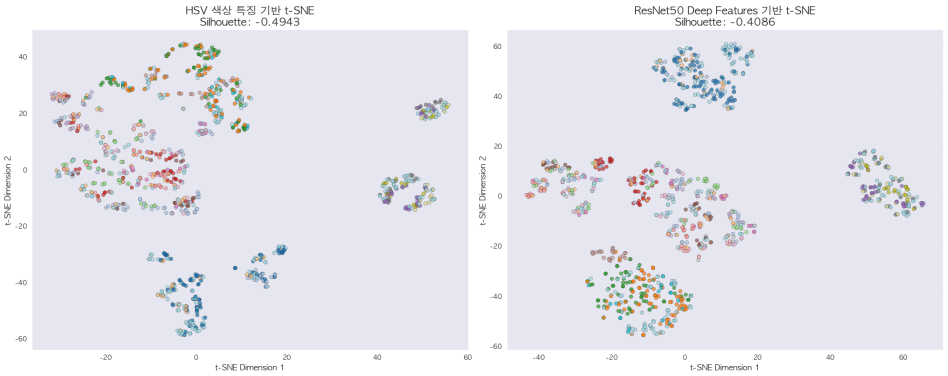
BBox(알약) 중심점 2D 히트맵 분석 결과, 4,500개가 넘는 모든 알약이 이미지 **좌측 하단의 극히 일부 영역에만 초밀집**되어 있다.

결론: 모델은 이미지 전체에서 객체를 '검출(Detection)'하는 어려운 문제를 푸는 것이 아니라, 항상 정해진 위치에 나타나는 물체를 '분류(Classification)'하는 쉬운 문제로 변질된 것이나 다름이 없다. 추측컨데 약의 오용을 방지하기 위해 폐쇄된 환경에서 촬영한 것 같다는 생각을 했다.

2-3. 발견 3: 극단적인 '특징' 편향 (t-SNE & Silhouette 분석)



- 처음에는 그냥 전반적으로 비슷하게 생긴 이미지들을 보고 비슷한 약들끼리 묶어봤다 데이터는 AI가 보기에 **구분이 가능한, 명확한 그룹들로 나뉘는듯 하다.(이미지를 그냥 전반적으로 보기때문에 텍스트 각인 같은 것도 포함된다)**
- 데이터가 복잡하지 않고 꽤 단순한듯 하다.



- 여기서 이제 색상(HSV)과 형태(ResNet50 임베딩, ResNet은 텍스트 각인은 무시한다 ResNet은 모양과 형태 그자체에 집중할 수 있다) 기반으로 군집화한 결과, 실루엣 점수 **-0.49**로 클래스 간 구분에 실패. 즉, 색상·형태만으로는 분류 불가능함을 확인했다.
- 대부분의 클래스가 뒤섞여 실제 라벨(73개)과 불일치한다. 일부 독특한 알약만 주변부에 '섬'처럼 분리되었으나, 이는 전체 문제 해결과 무관하다.
- **그런데 YOLO vs t-SNE에서는**
  - t-SNE(색상/형태)은 분류에 실패했고
  - YOLO은 mAP 0.99로 분류 성공했다
  - **이유:** YOLO는 색상·형태가 아니라 **알약 표면의 텍스트 각인(Imprint)** 같은 미세 특징(Fine-grained Feature)에 의존해 분류를 수행했다.

EDA 결론:

본 데이터셋은 배경, 조명, 위치, 색상 그리고 형태 등 모든걸 무시하는 상태에서, 텍스트 각인이라는 단일하고 뚜렷한 특징만 남아 있는 '쉬운 문제'였다. 따라서 mAP 0.99는 모델의 일반화 능력을 의미하지 않으며, 결국 모델은 알약을 '인식'한 것이 아니라 **깨끗한 배경에서 텍스트를 읽는 기계**로 학습된 것에 불과하다.

2-4. 발견 4: 데이터 '신뢰도' 문제

앞선 3대 편향 외에도, 데이터셋 자체의 신뢰도 문제 역시 심각하게 확인되었다.

- **라벨 누락:** 전체의 57% (850개)에서 파일명(K-ID)과 실제 라벨(BBox) 개수가 불일치하는 오류셋 발견
- **클래스 불균형:** 상위 20개 클래스가 전체 BBox의 약 80%를 차지하는 **Long-tail 분포** 확인
- **품질 오류:** BBox 경계 이탈(OOB), 중첩(High IoU) 등 **17개 불량 데이터** 식별 및 제거

결론: 원본 데이터셋은 그대로 활용하기 어려웠으며, **의사 레이블링(Pseudo-Labeling)과 정제 과정**이 필수적이었다.

3. 새로운 목표 및 데이터 파이프라인 구축

EDA 결과를 바탕으로, 프로젝트의 목표와 데이터 처리 체계를 다음과 같이 재설정하였다.

3-1. 신규 목표: '안정성(Loss)'과 '효율성(Cost)' 탐색

- **안정성(Stability):** `Val Box Loss`, `Val Cls Loss`, `Val DFL Loss`를 가장 낮고 안정적으로 유지하는 모델 탐색
- **효율성(Cost):** mAP 0.99 및 안정적 Loss를 달성하는 데 필요한 최소 학습 시간(Epochs)과 모델 크기(Params) 최적화
- mAP 점수도 어느정도 조건에 맞으면 높은 조합을 고려해본다.

3-2. 데이터 파이프라인: '1472 통합 데이터셋' 구축

1. **정제:** 원본 1,489개 중 품질 오류 17개 제거 → 1,472개 확보
2. **분류:** 1,472개를 [클린셋 632개]와 [오류셋 840개]로 구분
3. **복원:** 오류셋 840개를 Roboflow 기반 의사 레이블링으로 복원
4. **통합:** 클린셋 632개 + 복원 오류셋 840개 → 총 1,472개 통합 데이터셋 완성
5. **분할:** `Train (70%) / Val (15%) / Test (15%)`로 계층 분할 후 실험 시작

4. 실험 1: 모델 Cost 분석 (담당: 이경식)

결론 요약: "YOLOv8n이 가장 효율적인(Cost-Effective) 모델"

앞선 EDA 결과를 통해 mAP 0.99 달성이 어렵지 않음을 확인하였다. 이에 따라 본 실험에서는 mAP 0.99에 도달하기 위한 최소 학습 비용(시간, 모델 크기)을 탐색하는 것을 1차 목표로 설정하였다.

4-1. 실험 환경 및 공통 하이퍼파라미터

- **H/W 환경:** Windows 11, NVIDIA GeForce RTX 3060Ti
- **S/W 환경:** Python 3.13.9
- **공통 하이퍼파라미터:**
- `data` : `CombinedDataset/data.yaml` (총 1,472개 샘플)
- `epochs` : 70 (기준)
- `imgsz` : 640
- `batch` : 16
- `optimizer` : AdamW
- `lr0` : 0.001
- `patience` : 10

4-2. 실험 결과: mAP@75 0.99 도달 시간

Model	Model Size	0.99 최초 도달 Epoch	0.99 최초 도달 시간
yolo8	**n**	**32**	**3분 44.5초**
yolo8	s	38	3분 52.9초
yolo8	m	38	7분 29.9초
yolo9	t(=n)	32	5분 46.2초
yolo9	s	32	7분 17.9초
yolo9	m	32	1시간 8분 46.6초
yolo10	n	50	4분 43.6초
yolo10	s	50	10분 21.2초
yolo10	m	50	1시간 14분 19.5초
yolo11	n	실패 (0.97 종료)	-
yolo11	s	실패 (0.97 종료)	-
yolo11	m	실행 안 함	-

4-3. 결론

- **성능 비교 결과:** `yolov8`과 `yolov9` 계열이 전반적으로 가장 우수한 성능을 보였다.

- **비용 효율성:** 특히 `yolov8n (nano)` **모델은 3분 44.5초 만에 mAP 0.99에 도달**하여, 학습 속도와 자원 효율성 측면에서 가장 뛰어난 결과를 기록하였다.
- `yolov9` **분석:** `t(=n)/s/m` 전 모델이 32 Epoch 내에 mAP 0.99를 달성했으나, `m` 모델의 경우 학습 시간이 1시간 이상 소요되었다. 따라서 고사양 환경에서는 `yolov9`의 활용 가능성이 존재한다.
- `yolov11` **분석:** `n`, `s` 모델 모두 mAP 0.97에서 학습이 종료되어, 본 과제(경구약제 이미지 검출)에는 적합하지 않은 것으로 판단된다.
- `yolov12` 도 실험을 했는데 s를 하다가 파이썬 자체가 다운이 되어서 실험의 의미가 없어져서 `yolov12` 는 실험에서 아예 제외가 되었다.
- **최종 결론: 비용 효율성과 안정성을 종합적으로 고려할 때,** `yolov8n` **을 Baseline 모델로 채택**한다.

5. 실험 2: Baseline Optimizer 선정 (담당: 이승완)

결론 요약: "mAP는 2개 옵티마이저를 제외하고 모두 0.99, 안정성(Loss)에서 `AdamW` **가 압승"**

실험 1에서 `YOLOv8n`이 가장 효율적인(Cost-Effective) 모델로 선정됨에 따라, 2단계에서는 해당 모델의 **안정성을 극대화할 최적 옵티마이저**를 탐색하였다.

5-1. 실험 환경

- **Model:** `yolov8n`
- **Epochs:** 70
- **Augmentation:** `False` (옵티마이저 순수 성능 비교를 위해 증강 OFF)
- **Optimizers Tested** (각 옵티마이저에 기본값이라 생각되는 학습률들로 비교)
  - `SGD` (lr=0.01)
  - `RMSprop` (lr=0.001)
  - `Adam` (lr=0.001)
  - `Nadam` (lr=0.001)
  - `AdamW` (lr=0.001)

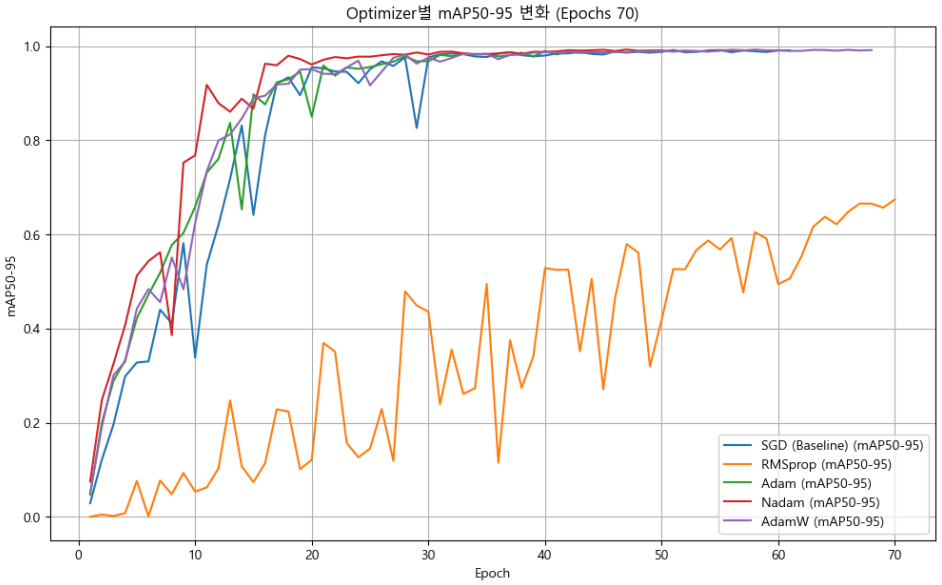
5-2. 실험 결과 요약

각 옵티마이저의 `best.pt` 모델(mAP50-95 기준) 성능 요약

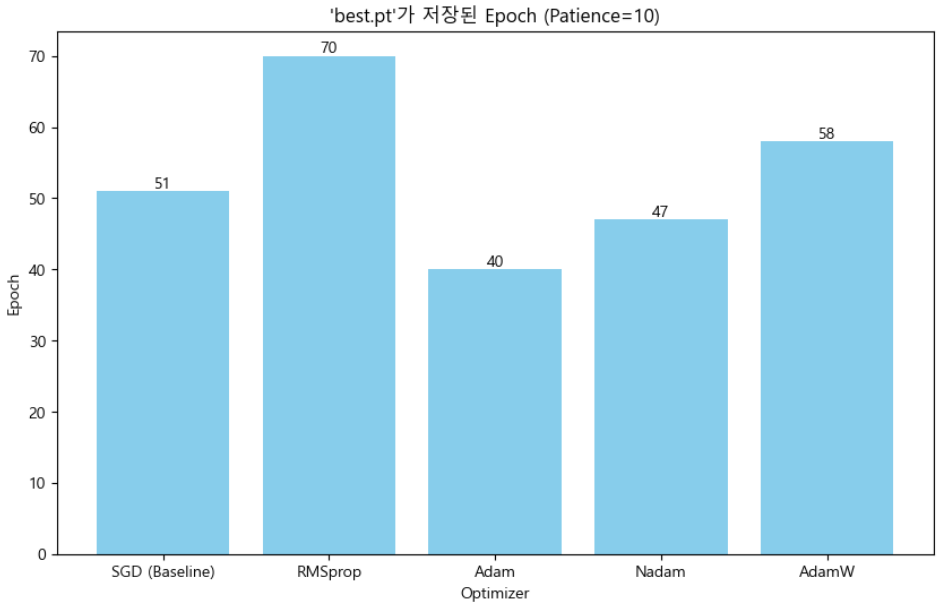
옵티마이저	Best Epoch	mAP50-95	Val Box Loss	Val Cls Loss	Val DFL Loss	Train Box Loss	총 시간 (s
AdamW	58	**0.9925**	**0.1543**	**0.1454**	**0.7621**	0.2295	1539.21
Nadam	47	**0.9926**	0.1716	0.1702	0.7661	0.2670	**1317.55**
SGD	51	0.9916	0.1718	0.2045	0.7736	0.2596	1418.18
Adam	40	0.9896	0.1882	0.2107	0.7716	0.2728	1122.92
RMSprop	70	0.6741	0.1886	0.8212	0.7874	**0.2177**	1608.83

5-3. 심층 분석

① 성능(mAP) 및 학습 속도

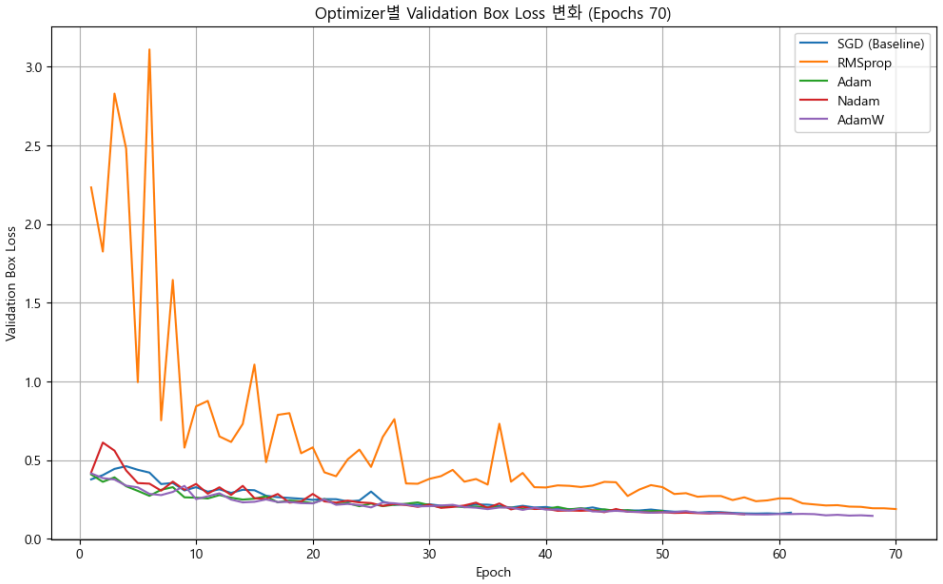


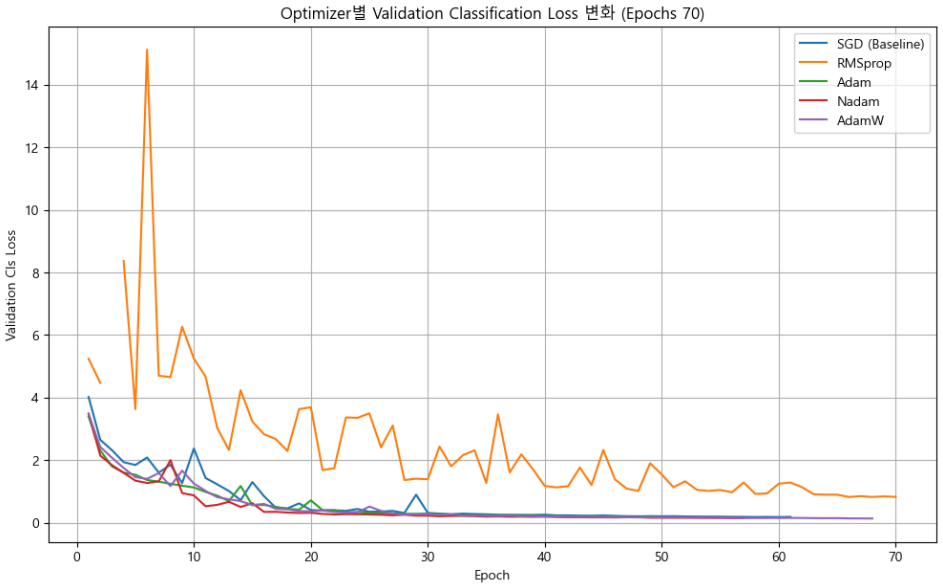
- `RMSprop, Adam`을 제외한 4개 옵티마이저 모두 mAP 0.99 이상을 무난히 달성하였다.
- `Nadam`과 `AdamW`가 각각 0.9926, 0.9925로 가장 높은 수치를 기록했으며, 성능 차이는 미미했다.

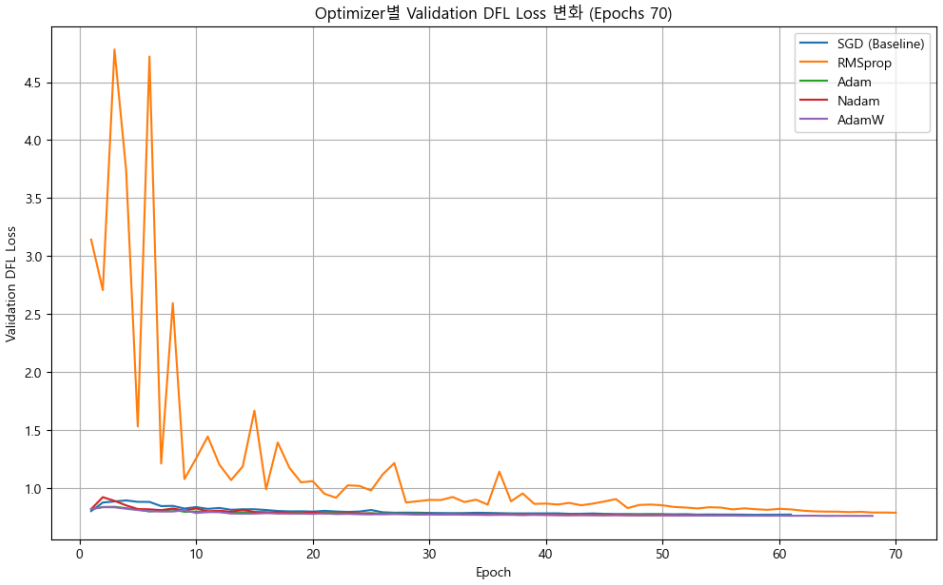


- 학습 속도 측면에서는 `Adam`과 `Nadam`이 가장 빠른 결과를 보였으나, 안정성(Validation Loss)도 확인을 해보았다.

② 핵심: 안정성 (Validation Loss) 비교


Optimizer별 Validation Box Loss 변화 (Epochs 70)
SGD (Baseline)
RMSprop
Adam
Nadam
AdamW
Validation Box Loss
Epoch


Optimizer별 Validation Classification Loss 변화 (Epochs 70)
SGD (Baseline)
RMSprop
Adam
Nadam
AdamW
Validation Cls Loss
Epoch

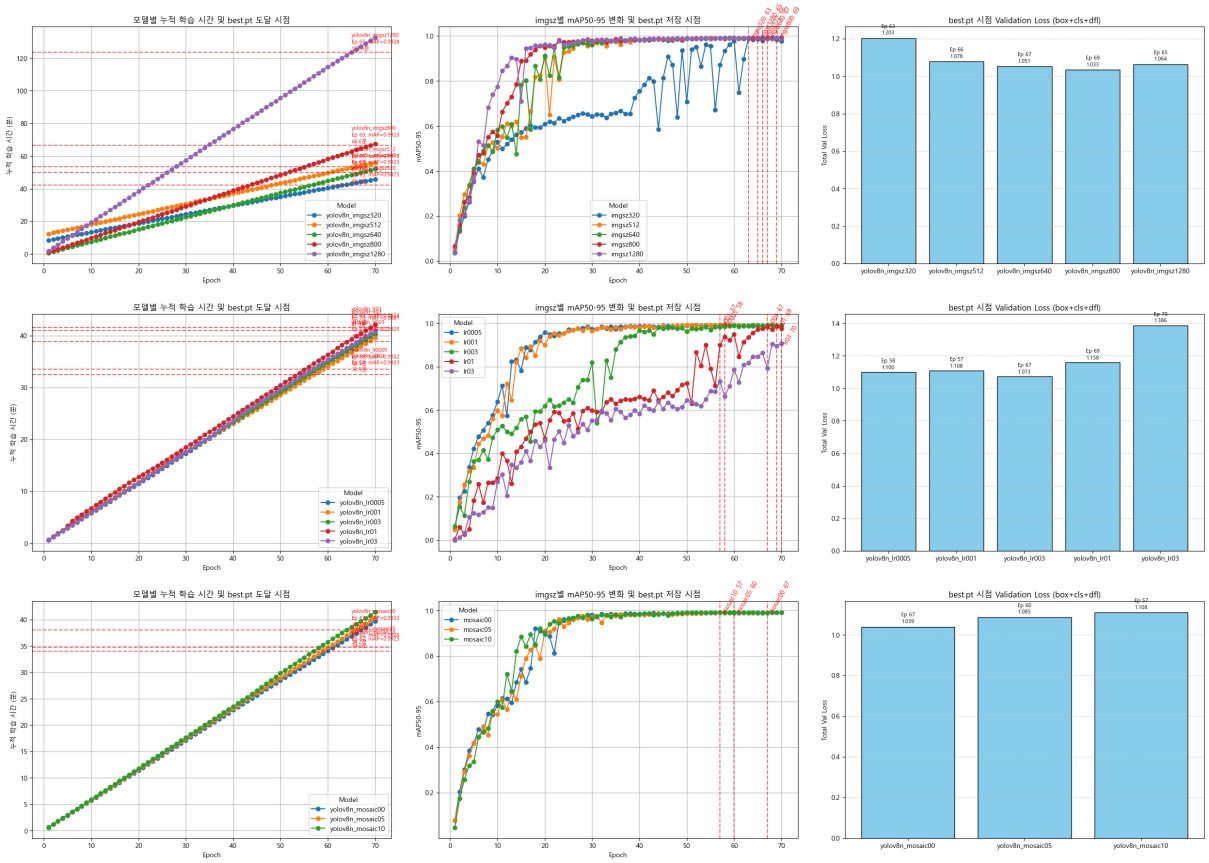

Optimizer별 Validation DFL Loss 변화 (Epochs 70)
SGD (Baseline)
RMSprop
Adam
Nadam
AdamW
Validation DFL Loss
Epoch

- **AdamW**가 3개 Validation Loss 지표 모두에서 가장 낮은 수치를 기록하며 안정성을 입증하였다.
- **Val Box Loss (위치 정확도):** `AdamW` = 0.1543 (가장 낮고 안정적). 반면 `RMSprop`은 학습 내내 불안정성을 보임.
- **Val Cls Loss (분류 정확도):** `AdamW` = 0.1454로, Nadam(0.1702), SGD(0.2045) 대비 확연히 우수. `RMSprop`은 0.8212로 사실상 분류 실패.
- **Val DFL Loss (경계 정확도):** `AdamW` = 0.7621로 가장 낮은 수치를 기록.

5-4. 결론

- **최종 채택:** 팀 Baseline 옵티마이저로 `AdamW`를 선정한다.
- `Nadam`은 학습 속도가 약 17% 빠르고 mAP 수치도 유사했으나, **AdamW가 Val Box/Cls/DFL Loss 모두에서 최저값을 기록**하며 안정성 측면에서 압도적 우위를 보였다.
- `RMSprop`는 본 데이터셋 특성상 부적합함을 확인하였다.
- **향후 모든 실험은 `YOLOv8n` + `AdamW` 조합을 Baseline으로 통일하여 진행한다.**

6. 실험 3: 하이퍼파라미터(`imgsz(이미지 크기)` , `lr0(학습률)` , `mosaic(모자이크 증강)`) 튜닝 (담당: 오현민)



결론 요약: mAP 기준(lr=0.0005, mosaic=0.5)과 Loss 기준(lr=0.003, mosaic=0.0)의 상충되는 두 조합을 도출. 최종 Test 셋에서 두 조합을 모두 검증하기로 결정.

Baseline 모델(`yolov8n`)과 옵티마이저(`AdamW`) 확정 후, 모델의 안정성(Loss)과 효율성(Cost)을 극대화하기 위한 3가지 하이퍼파라미터(`imgsz(이미지 크기)` , `lr0(학습률)` , `mosaic(모자이크 증강)`) 튜닝을 진행했다.

- **공통 Baseline:** `yolov8n` , `AdamW` , `epochs=70` , `batch=16`

6-1. `imgsz` (이미지 크기) 실험

- **목표:** 이미지 크기가 모델 안정성(Loss)과 학습 비용(시간)에 미치는 영향 분석.

Model	mAP50-95 (참고)	Total Val Loss	Time to Best (min)	Notes
yolov8n_imgsz320	0.9875	1.2033	42.12	Loss 높음 (불안정)
yolov8n_imgsz512	0.9918	1.0781	53.49	안정화 시작
yolov8n_imgsz640	0.9923	1.0515	**49.89**	**Loss-Cost 밸런스 최적**
yolov8n_imgsz800	**0.9923**	**1.0332**	66.59	**가장 안정적 (Loss 최저)**
yolov8n_imgsz1280	0.9928	1.0637	123.64	Loss 증가, 비효율

• **결론:**

- `imgsz=800` 이 **Total Loss 1.0332**로 가장 안정적인(Loss가 가장 낮은) 모델로 확인되었다.
- `imgsz=640` 은 Loss(1.0515)는 800 대비 근소하게 높지만, 학습 시간이 **30%** 더 빨라 '비용 대비 효율(Efficiency)'이 가장 뛰어났다.
- mAP가 가장 높았던 `imgsz=1280` 은 오히려 Loss가 증가하고(1.0637) 학습 시간이 2.5배 소요되어 비효율적이었다.

6-2. `lr0` (Learning Rate) 실험

• **목표:** `AdamW`의 최적 안정화(Loss 최소화) 학습률 탐색.

Model	mAP50-95 (참고)	Total Val Loss	Time to Best (min)	Notes
yolov8n_lr0005	0.9932	1.1000	33.55	Loss 높음 (mAP만 높음)
yolov8n_lr001	0.9923	1.1083	32.45	Baseline (불안정)
yolov8n_lr003	**0.9925**	**1.0730**	38.85	**가장 안정적 (Loss 최저)**
yolov8n_lr01	0.9859	1.1579	41.50	불안정
yolov8n_lr03	0.9081	1.3865	40.98	발산, 실패

• **결론:**

- `lr=0.0005`: 가장 높은 mAP(0.9932)와 가장 빠른 수렴 속도(33.55분)를 보였으나, Total Loss(1.1000)는 상대적으로 높았다.
- `lr=0.003`이 Total Loss **1.0730**을 기록하며 **가장 안정적인(Loss 최저) 학습**을 보였다.
- `lr=0.01` 이상에서는 Loss가 급증하며 학습이 불안정해졌다.

6-3. `mosaic` (모자이크 증강) 실험

• **목표:** EDA 결과(극단적 편향)를 바탕으로 Augmentation이 안정성(Loss)에 미치는 영향 분석.

Model	mAP50-95 (참고)	Total Val Loss	Time to Best (min)	Notes
yolov8n_mosaic00	**0.9922**	**1.0388**	38.00	**가장 안정적 (Loss 최저)**
yolov8n_mosaic05	0.9926	1.0852	34.73	Loss 증가 (mAP만 높음)
yolov8n_mosaic10	0.9923	1.1083	34.02	Loss 증가 (과한 증강)

• **결론:**

- `mosaic=0.5`: 가장 높은 mAP(0.9926)와 빠른 수렴 속도를 보였으나, Total Loss(1.0852)가 증가했다.
- `mosaic=0.0` (증강을 끈 상태)가 Total Loss가 **1.0388**로 더 안정적이었다.
    - EDA(2-2)에서 확인했듯, 우리 데이터는 '극단적인 공간 편향'(좌측 하단 밀집)을 가진다. 이처럼 이미 공간적 특성이 고정된 데이터에 `mosaic`와 같은 **'기하학적/공간적' 증강**을 추가하는 것은 불필요한 노이즈로 작용하여 오히려 안정적인 수렴을 방해하는 것으로 판단된다.

6-4. 하이퍼파라미터 종합 결론

실험 결과, 'mAP/속도' 기준과 'Loss 안정성' 기준 간에 상충되는 두 가지 최적 조합이 도출되었다.

- **[조합 (Loss/안정성/효율 기준)]:** `imgsz640 + lr0=0.003 + mosaic=0.0`
    - `imgsz=640`은 Loss(1.0515)는 800 대비 근소하게 높지만, 학습 시간이 **30%** 더 빨라 '비용 대비 효율(Efficiency)'이 가장 뛰어나서 선정이 되었다.

7. 실험 4: Augmentation 튜닝 (담당: 최준영)

결론 요약: `hsv_s = 0.2 + hsv_v = 0.4` **조합, 최고의 효율 및 안정성 균형 달성**

- **공통 Baseline:** `yolov8n`, `AdamW`, `lr=0.003`, `imgsz=640`, `mosaic=0.0`

7-1. 명도(Value) 증강 (`hsv_v`)

가장 기본적인 조명 변화(명도)에 대한 안정성(Loss)과 성능(mAP)의 균형점을 탐색했다.

실험	명도 (hsv_v)	Best mAP50-95	Val Loss (최소)	Best Epoch	분석
v1	0.1	0.9769	1.1467	30	효과 미미
v2	0.2	0.9838	1.1434	34	약간의 성능 개선, 수렴 지연
v3	**0.3**	0.9828	1.1483	30	성능 약간 하락, 수렴 속도 개선
v4	0.4	0.9847	**1.1366**	29	Loss/mAP 균형점
v5	0.5	0.9831	1.1320	38	성능 포화, 수렴 지연

- **결론:**
  - `v=0.5` (v5)가 가장 낮은 Loss(1.1320)를 기록했으나, mAP가 하락하고 수렴이 지연(Epoch 38)되는 성능 포화 현상을 보임.
  - 명도 `v=0.4` (v4)에서 안정적인 Loss(**1.1366**)와 높은 mAP(0.9847) 그리고 가장 빠른 수렴속도(29)를 보여, `hsv_v=0.4` **를 후속 컬러 실험의 Baseline으로 확정했다.**

7-2. 색상(Hue) 및 채도(Saturation) 증강

안정적인 명도 `v=0.4`를 고정한 상태에서, 색조(Hue)와 채도(Saturation)가 성능과 안정성에 미치는 영향을 비교했다.

실험	(H)	(S)	(V)	Best mAP50-95	Val Loss (최소)	Best Epoch	분석
v6	0.1	0.0	0.4	0.9779	1.1154	63	mAP 하락 지연
v7	0.2	0.0	0.4	0.9856	1.1007	62	조금의 성 여전한 수
v8	0.0	0.1	0.4	0.9842	1.1458	34	mAP/Los 우수
v9	**0.0**	**0.2**	**0.4**	**0.9857**	1.1643	23	**mAP 최고 수렴 속도 최적)**
v10	0.0	0.3	0.4	0.9790	1.2026	18	mAP 급격

- **결론:**
  - **색조(Hue) 증강 (v6, v7):** `v7(H=0.2)`이 **mAP 0.9856** (v9와 거의 동급) 및 Val Loss 1.1007 (전체 1위)을 기록, '성능'과 '안정성'을 모두 잡았음. 단, 수렴 속도(Epoch 62)가 매우 느려 '효율성'이 떨어짐. `v6`는 mAP가 0.97대로 하락함.
  - **채도(Saturation) 증강 (v8, v9, v10):** `v10(S=0.3)`은 mAP가 0.97대로 하락하고 Val Loss(1.2026)가 급증하여 가장 불안정한 '실패' 모델로 확인됨.
  - `v9 (s=0.2)`는 최고 mAP(0.9857)와 가장 빠른 수렴 속도(Best Epoch 23)를 기록하여 '효율성' 측면에서 압도적으로 우수했음.
  - `v7 (h=0.2)`은 **Val Loss가 1.1007**로 모든 실험 중 가장 낮아 '안정성'은 가장 높았으나, 수렴 속도(Epoch 62)가 `v9` 대비 **2.7배 느려** '효율성'이 급격히 떨어짐.
  - '최고 효율'(v9)을 선택할 것인가, '최고 안정성'(v7)을 선택할 것인가의 Trade-off 관계가 확인됨.
    - 결과적으로 최고 효율의 v9 선택

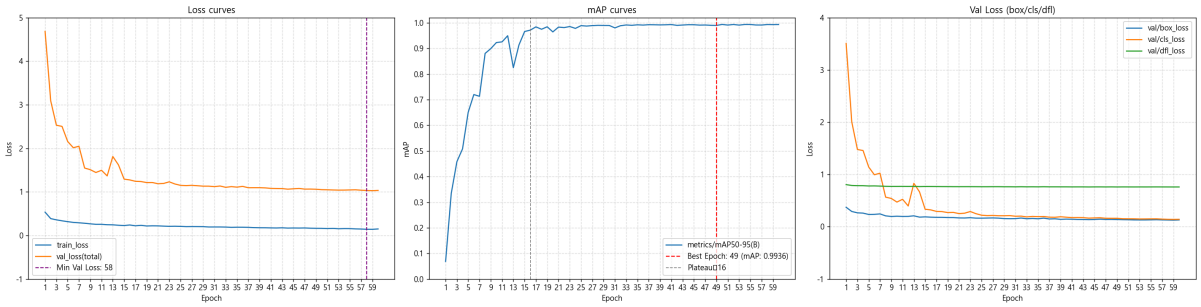
7-3. 기하학(Geometric) 증강

마지막으로, 최종 컬러 조합(v9)에 회전, 이동 등 기하학 증강을 추가하여 강건성(Robustness) 한계를 테스트했다.

실험	증강	Best mAP50-95	Val Loss (최소)	Best Epoch	분석
v9	컬러 (`s=0.2, v=0.4`)	**0.9857**	**1.1643**	23	기준 모델
v11	**v9 + 기하학**	0.9485	1.3554	39	**mAP 3.7%p 급락 (실패)**

• **결론:** 기하학 증강(v11)은 **mAP가 0.9485로 치명적인 수준(-3.7%p)으로 하락한다**. 이는 EDA(2-2)에서 발견한 '극단적 공간 편향' 데이터에 과적합된 모델이 위치/각도 변화에 매우 취약함을 의미한다.

7-4. Augmentation 최종 결론



`v9 (hsv_s=0.2, hsv_v=0.4)` 조합을 최종 Augmentation Baseline으로 채택했다.
주요 근거:

1. **[실패 검증]** `v11` (기하학 증강)은 mAP 3.7%p 급락(0.9485)을 초래했다. 이는 EDA(2-2)에서 발견한 '극단적 공간 편향'에 과적합된 모델이 위치/각도 변화에 매우 취약함을 증명하며 기하학 증강은 최종 탈락했다.
2. **[효율/성능 검증]** `v9`가 최고 mAP(0.9857)와 가장 빠른 수렴 속도(Best Epoch 23)를 동시에 달성하며 '효율성' 측면에서 압도적으로 우수했다.
3. **[안정성 Trade-off]** `v9`의 Val Loss(1.1643)는 `v7` (1.1007)보다 높게 측정되었다. 즉, `v7`이 '최고 안정성'을 달성했으나, 수렴 속도(Epoch 62)가 `v9` 대비 2.7배 느려 '효율성'이 급격히 저하되었다. 따라서 **'최고 효율'을 선택**하고 Val Loss(1.1643)를 감수하는 것이 팀 목표에 더 부합한다고 판단했다.
4. EDA(2-4)에서 t-SNE(실루엣 -0.49)는 '색상/형태'만으로 분류에 실패했으나 YOLO(mAP 0.99)는 '텍스트 각인'으로 분류에 성공했음을 확인하였다. `v9`의 채도(S)와 명도(V) 증강은 '텍스트 각인' 구분에 핵심인 **'대비(Contrast)'를 강화**시켜, 모델이 이 미세 특징을 더 빠르고 안정적으로 학습하도록 돕는 최적의 조합임을 확인했다.

8. 최종 검증(Test set 사용) (담당: 최준영)

[실험 1 결론]: '속도/효율' 기준 최적 조합은 `yolov8n`

+

[실험 2 결론]: 'Loss/안정성**' 기준 최적 조합은** `AdamW`

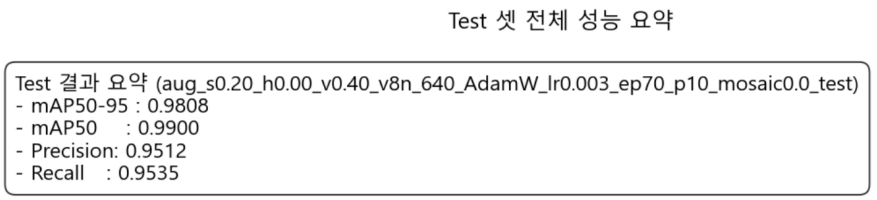
+

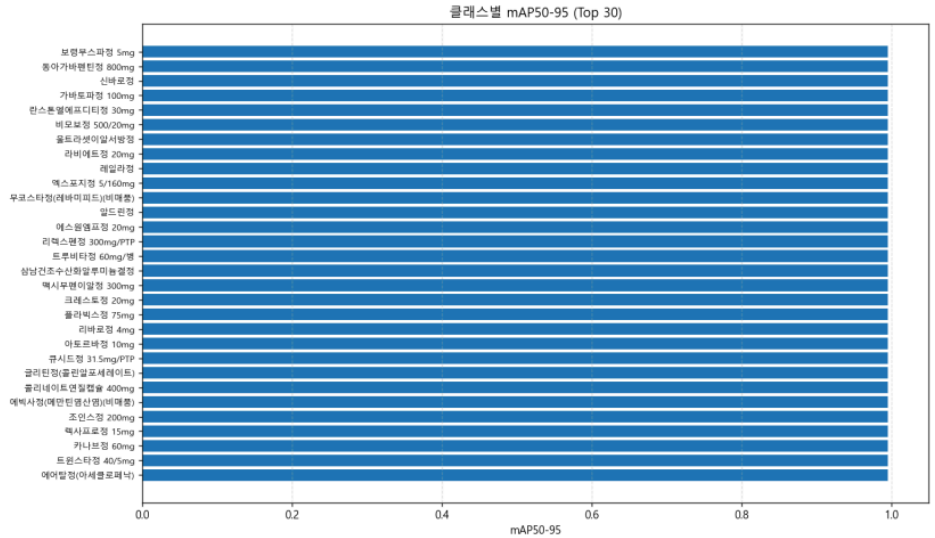
[실험 3 결론]: 'Loss/안정성/효율' 기준 최적 조합은 `imgsz=640` `lr=0.003` , `mosaic=0.0`

[실험 4 결론]: '효율/성능' 기준 최적 조합은 `hsv_s(채도)=0.2` , `hsv_v(명도)=0.4`

최종 조합

```
(
    data=yaml_path_combined,
    epochs=70,
    patience=10,
    imgsz=640,
    batch=16,
    device=0,
    project=exp_dir,
    name=experiment_name,
    exist_ok=True,
    optimizer='AdamW',
    lr0=0.003,          # 기존 0.003 유지 (v9)
    augment=True,     # Augmentation on
    mosaic=0.0,       # mosaic 끔
    hsv_s=0.2,        # 채도(S) 0.2
    hsv_v=0.4,        # 명도(V) 0.4
    workers=0,
)
```





- **성능 검증 (가설 확인):**
  '안정성'과 '효율성'을 기준으로 도출된 최종 조합(`lr=0.003, mosaic=0.0, s=0.2, v=0.4`)은, 15% Test 셋에서 **mAP50-95 0.9808**이라는 높은 점수를 달성했다.
  이는 Test 셋 역시 Train 셋과 동일한 '3대 편향'을 공유하고 있음을 의미하며, "mAP 0.99는 함정" 데이터셋 자체에 문제가 있다라는 EDA의 핵심 가설을 최종적으로 입증한다.
- **신뢰도 검증 (과적합 확인):**
  클래스별 mAP(Top 30) 분석 결과, 대부분의 주요 클래스가 mAP 1.0에 근접했다. 이는 모델이 '일반 특징'이 아닌 '텍스트 각인'이라는 '미세 특징'을 100% 암기(과적합)했으며, Test 셋에서도 이 암기한 특징을 성공적으로 재현했음을 보여준다.
- **가설 검증 (전략 유효성):**
  EDA(2-3)에서 제기했던 '텍스트 각인' 가설이 최종 입증되었다. '공간적 증강(mosaic=0.0)'을 배제하고 '텍스트 대비(contrast)'를 강화하는 '색상 증강(HSV)'을 적용한 우리의 전략이, 이 편향된 데이터셋 내에서 mAP를 달성하는 가장 유효한 접근법이었음이 증명되었다.