

**SK네트웍스 Family AI 과정 13기  
 데이터 전처리 인공지능 학습 결과서**



| **산출물 단계** | 데이터 전처리 |
| --- | --- |
| **평가 산출물** | 인공지능 학습 결과서 |
| **제출 일자** | 08/10 |
| **깃허브 경로** | https://github.com/SKNETWORKS-FAMILY-AICAMP/SKN13-FINAL-2TEAM |
| **작성 팀원** | 민경재 |

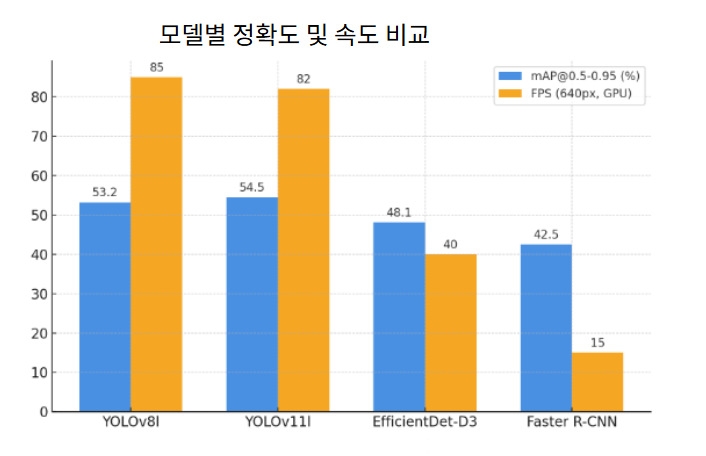
**1. 모델 개요**

* 모델명: YOLOv8-Large (yolov8l)
* 참고 문서: [Ultralytics YOLOv8 공식 문서](https://docs.ultralytics.com/ko/models/yolov8/)
* YOLOv8 주요 특징 (Ultralytics 공식 문서 기준)
  + 향상된 백본/넥 구조 및 앵커 프리 탐지 헤드 적용으로 정확도·속도 균형 최적화
  + 다양한 컴퓨터 비전 과제(탐지, 분류, 분할, 자세 추정 등) 지원
  + 모델 성능 지표 (COCO 640 기준)
    - YOLOv8l mAP@0.5-0.95 ≈ 52.9%
    - 빠른 추론 속도 (CPU/GPU 모두 균형)
    - 다양한 크기 옵션(n,s,m,l,x) 제공
* 모델 선정 이유 :

본 프로젝트에서는 **대규모(5~10만 장) 의류 이미지**를 대상으로 한 객체 탐지에서, 정확도·속도·안정성의 균형이 필수적이었습니다.

이에 다양한 객체 탐지 접근 방식을 대표하는 최신 단일 단계 모델(YOLOv8I, YOLOv11I)과 경량·효율성을 강조한 모델(EfficientDet-D3) 그리고 2단계 검출기(Faster R-CNN)을 비교 분석한 결과, **YOLOv8l**을 최종 선정하였습니다.

| **모델** | **mAP@0.5-0.95(%)** | **추론 속도** | **주요 장점** | **주요 단점** |
| --- | --- | --- | --- | --- |
| YOLOv8l | 53.2 | 85 | 높은 정확도와 속도, Anchor-free 구조, 다양한 크기 객체 대응 우수, 배포 용이 | 모델 크기가 YOLOv8s/m보다 큼 |
| YOLOv11l | 54.5 | 82 | 최신 아키텍처 적용, 일부 벤치마크에서 mAP개선, 데이터셋 일반화 성능 강화 | 실무 적용 레퍼런스 적음, 초기 버전으로 장기 안정성·호환성 검증 부족 |
| EfficientDet-D3 | 48.1 | 40 | 효율적인 파라미터 사용, 작은 모델 크기 | 속도 느림, 소형 객체 검출 약함 |
| Faster R-CNN | 42.5 | 15 | 높은 정밀도, 클래식한 안정성 | 속도 매우 느림, 대규모 데이터 처리 부적합 |



YOLOv8-L 선택 근거 :

* 정확도와 속도의 최적 균형
  + YOLOv8l은 mAP@0.5~0.95가 **53.2%**로 YOLOv11l과 근소한 차이지만, 처리 속도(FPS)는 더 높아 대규모 이미지 배치 처리에 유리
* 검증된 안정성
  + YOLOv8l은 다양한 산업·연구에서 실무 적용 사례가 많아, 장기 운영 시 예측 불가능한 호환성·안정성 문제 가능성이 낮음
* 서비스 배포 및 호환성
  + Ultralytics 생태계와 완벽 호환, TensorRT/ONNX 변환 검증 완료 사례 다수
  + Roboflow, 벡터 DB 구축 파이프라인과의 연동이 용이
* 타 모델 대비 제약 적음
  + YOLOv11l은 초기 버전 특성상 API 변환 과정에서 호환성 문제가 보고됨
  + EfficientDet-D3와 Faster R-CNN은 속도·처리 효율성에서 대규모 데이터 처리에 한계

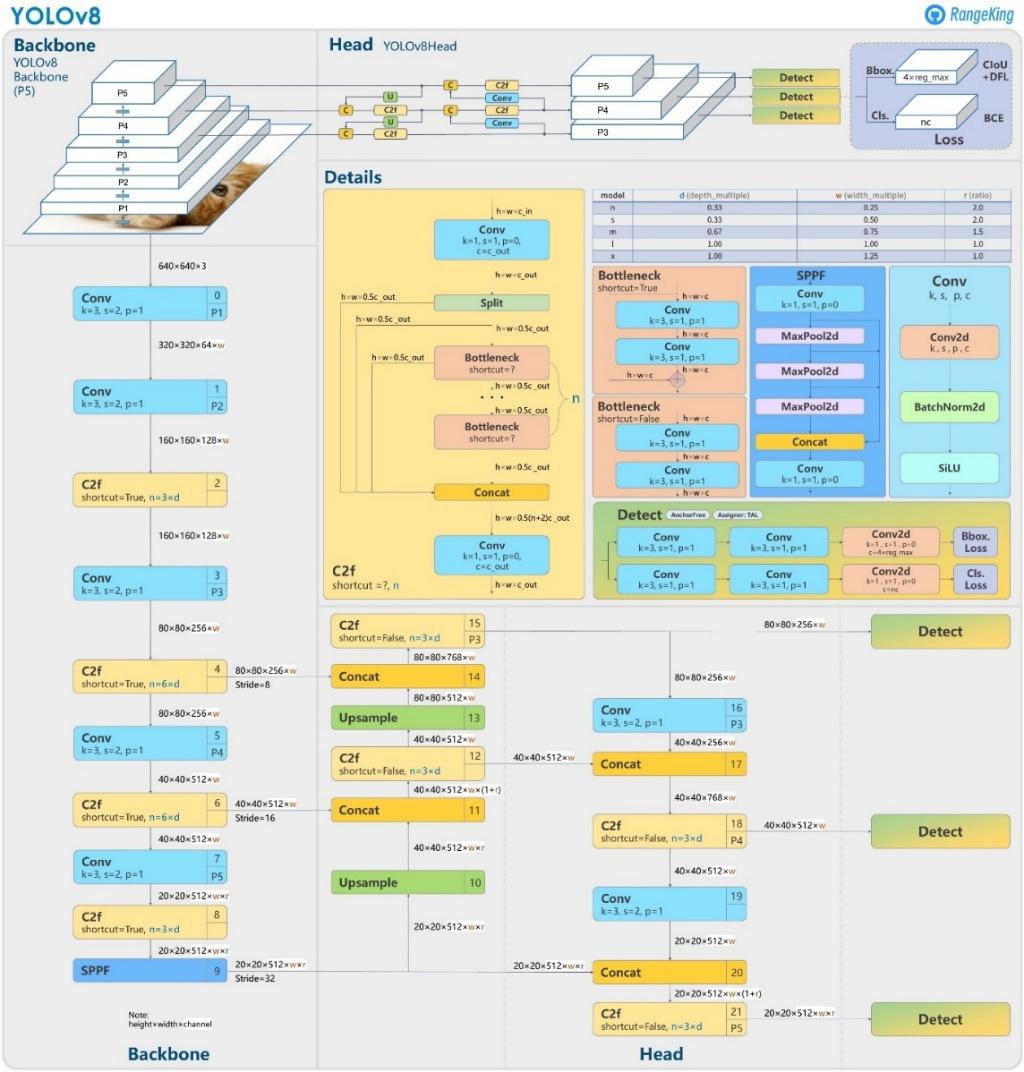
결론 :

YOLOv8l은 본 프로젝트의 요구사항인 **대규모 데이터 안정 처리 + 실시간 추론 + 다양한 객체 크기 검출**에 가장 적합한 모델로 판단되어 최종 선정

**2. 모델 구조 및 아키텍처**

2.1 모델 아키텍처 개요

* 입력 이미지 → Backbone(CSPDarknet) → Neck(PAN-FPN) → Detection Head(Anchor-free, Class & Box 예측)



2.2 구성 요소 설명

| 계층명 | 역할 | 구성 요소 |
| --- | --- | --- |
| Backbone | 특징 추출 | CSPDarknet 구조, Conv + C2f 모듈 |
| Neck | 다중 스케일 피처 통합 | PAN-FPN |
| Detection Head | 객체 탐지 | Anchor-free 구조, Bounding box 회귀 및 클래스 확률 예측 |

**3. 학습 설정 및 하이퍼파라미터**

3.1 1차 학습 (Initial Training)

| **항목** | **값** |
| --- | --- |
| 학습 데이터 수 | 4184 |
| 검증 데이터 수 | 400 |
| 테스트 데이터 수 | 200 |
| 에폭(Epoch) 수 | 150 |
| 배치 크기 (Batch Size) | 64 |
| 학습률 (Learning Rate) | 기본값 사용(약 0.01) |
| 옵티마이저 | AdamW (YOLOv8 기본) |
| 손실 함수 | YOLOv8 내장 손실 (Box loss, Obj loss, Class loss) |
| 조기 종료 기준 | validation loss 15 epoch 동안 개선 없을 시 종료 |
| 체크포인트 저장 주기 | 10 epoch마다 저장 |
| 저장 경로 | runs/train/yolov8l\_custom |

3.2 추가 학습 (Fine-tuning)

| **항목** | **값** |
| --- | --- |
| 학습 데이터 수 | 1545 |
| 검증 데이터 수 | 150 |
| 테스트 데이터 수 | 77 |
| 에폭(Epoch) 수 | 50 |
| 배치 크기 (Batch Size) | 64 |
| 초기 학습률 (Learning Rate, lr0) | 0.001 |
| 옵티마이저 | AdamW (YOLOv8 기본) |
| 손실 함수 | YOLOv8 내장 손실 (Box loss, Obj loss, Class loss) |
| 조기 종료 기준 | validation loss 15 epoch 동안 개선 없을 시 종료 |
| 체크포인트 저장 주기 | 10 epoch마다 저장 |
| 저장 경로 | runs/train/yolov8l\_custom\_ft |

* 결과 모델: best\_ft.pt (압축본: best\_ft.zip)

**4. 학습 결과 및 성능 평가**

| 모델 | Epoch | mAP@0.5 | mAP@0.5:0.95 | Precision | Recall |
| --- | --- | --- | --- | --- | --- |
| 1차 학습 (best.pt) | 103 | 0.974 | 0.837 | 0.918 | 0.960 |
| 추가 학습 (best\_ft.pt) | 28 | 0.905 | 0.766 | 0.825 | 0.891 |

4.1 분석

* 정확도(mAP)
  + 1차 학습 모델이 mAP@0.5와 mAP@0.5:0.95 모두에서 높은 값을 기록
  + 추가 학습 후 성능이 전반적으로 하락 (mAP@0.5 약 6.9%, mAP@0.5:0.95 약 7.1% 감소)
* 정밀도(Precision)
  + 1차 학습: 0.918 → 추가 학습: 0.825 (약 9% 감소)
* 재현율(Recall)
  + 1차 학습: 0.960 → 추가 학습: 0.891 (약 7% 감소)
* 원인 추정
  + **데이터 클래스 불균형**: 추가 학습 데이터셋에서 클래스별 분포가 균일하지 않아 특정 클래스의 학습이 상대적으로 부족했을 가능성이 큼. 예를 들어, 일부 클래스 (skirt&pants) 는 이미지 수가 50장 이하로 다른 클래스 대비 매우 적었음.
  + **데이터 규모 및 분포 차이**: 추가 학습 데이터셋(1,545장)이 기존 학습 데이터보다 규모가 작고, 클래스 분포가 달라 모델의 균형 잡힌 학습이 어려웠을 가능성.
  + **일반화 성능 저하**: Fine-tuning 과정에서 기존 모델의 일반화 성능이 일부 손상되었을 가능성.
  + **빠른 수렴 및 과적합 가능성**: 학습률을 낮췄음에도 불구하고 Epoch 50 중 28에서 조기 종료된 것은 데이터 규모 대비 빠른 수렴이나 과적합이 발생했음을 시사함.

4.2 그래프 분석

(1) Loss·mAP 변화 추이

1차 학습

* Box Loss, Obj Loss, Cls Loss가 안정적으로 감소
* mAP 지표가 학습 후반부에 안정적으로 수렴

텍스트, 도표, 지도, 평면도이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

추가 학습

* 초기 수렴 속도가 빠르지만, mAP 상승 폭 제한
* Epoch 28에서 조기 종료 → 데이터 규모 대비 빠른 수렴

텍스트, 도표, 폰트, 라인이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

(2) Confusion Matrix

1차 학습

* 클래스별 오탐률이 낮고 전반적으로 높은 Recall 유지

텍스트, 스크린샷, 도표, 평행이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

추가 학습

* 일부 클래스에서 혼동 증가, Recall 감소 원인

텍스트, 스크린샷, 도표, 평행이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

(3) Precision-Recall Curve

1차 학습

* 모든 클래스에서 Precision-Recall 곡선이 높은 성능 유지

텍스트, 스크린샷, 도표, 라인이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

추가 학습

* 일부 클래스에서 곡선이 완만해지며 Precision·Recall 동시 하락

텍스트, 도표, 라인, 그래프이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

4.3 종합 결론  
추가 학습(Fine-tuning)은 새로운 데이터셋에 적응하려는 시도였으나, 전반적인 성능 지표가 하락하였다. 향후에는 기존 학습 데이터 일부와 신규 데이터를 혼합하여 학습의 안정성을 높이고, 각 클래스별 데이터 개수를 최소 2,000장으로 맞춰 클래스 불균형을 해소하는 동시에 전체 데이터셋 규모를 확대함으로써 성능 저하를 방지할 필요가 있다.

**5. 과적합/과소적합 대응**

| 기법 | 설명 | 적용 여부 |
| --- | --- | --- |
| Data Augmentation | Flip, Rotation, Mosaic 등 변환 적용 | O |
| Early Stopping | 검증 지표 개선 없을 시 학습 종료 | O |
| Learning Rate Scheduling | Plateau 시 자동 학습률 감소 | O |
| Dropout | CNN 구조 특성상 미적용 | X |
| 데이터 혼합 학습 | 기존 데이터 + 신규 데이터 병합 학습 | X (추후 고려) |

* 결과 : 추후 데이터셋 확보 후 데이터 혼합 학습을 포함하여 다시 학습 예정

**6. 하드웨어 및 환경 정보**

| 항목 | 값 |
| --- | --- |
| GPU | Runpod A100 PCIe |
| CUDA 버전 | 12.8 |
| PyTorch 버전 | 2.8.0 |
| YOLOv8 버전 | YOLOv8l |
| OS | Ubuntu 20.04 |

**7. 결론 및 향후 계획**

최종 선정 모델: YOLOv8l

활용 방안

* 유사도 측정 시 다른 분류의 옷이 포함되지 않도록 해당 클래스의 의류만 크롭
* 사용자가 입력한 사진도 동일하게 크롭하여 유사도 측정에 활용

향후 계획

1. 각 클래스별 최소 2,000장의 이미지를 확보하여 클래스 불균형 해소
2. YOLO 모델로 이미지에서 해당 클래스의 의류만 크롭
3. GPT-4o를 활용해 크롭된 이미지의 특성을 반영한 캡션 생성
4. 생성된 캡션과 크롭 이미지를 기반으로 FashionCLIP을 사용해 임베딩 추출
5. 추출된 임베딩을 활용해 이미지 간 유사도 계산

9. 변경 이력

| 변경한 사람 | 변경 날짜 | 변경 내용 |
| --- | --- | --- |
| 민경재 | 2025.08.08 | 초안 작성 |
|  |  |  |
|  |  |  |