
해파리 종 분류

DLthon



목차

1. 프로젝트 개요

- 팀 목표 소개

2. EDA

- EDA 항목 소개
- 각 EDA 과정 및 결과
- 데이터 전처리 전략

3. 데이터 전처리

- 배경 제거
- BLUE 해파리 데이터셋 정제
- 중복 개체 이미지 제거

4. 베이스라인 모델

- 베이스라인 모델 설계
- 전처리 데이터 성능 검증

5. 모델 성능 개선

- 여러 모델 테스트
- 최고 성능 모델 개선

6. 결론

CONTENTS

1. 프로젝트 개요



목표: 높은 정확도



주안점: EDA 및 전처리

프로젝트 목표와 핵심 전략

- ☑ 높은 정확도를 달성하기 위해, 데이터의 불균형, 클래스 간 유사도, 이미지 품질 등 다양한 요인을 고려한 접근 전략을 세웠습니다.
- ☑ EDA와 전처리는 모델의 성능을 좌우하는 중요한 단계입니다. 데이터를 올바르게 이해하고 정제함으로써, 학습에 적합한 구조를 만드는 것이 핵심 전략입니다.

1. 프로젝트 개요

CURRENT

2. EDA

9가지 EDA 항목

-  클래스 별 이미지 수
-  이미지 크기
-  이미지 확장자
-  샘플 이미지 확인
-  중복 이미지 탐지
-  이미지 유사도 - 색상 분석
-  이미지 유사도 - 형태 분석
-  이미지 유사도 - 텍스처 분석
-  이미지 품질

2.1 EDA 항목 소개

— 2. EDA

2.2 샘플 이미지 확인

EDA - 샘플 이미지 확인

확인

- 해파리 외의 미칠 다른 요소들이 있는가? (배경 등...)
- 특정 무늬나 형태로 구분할 수 있는 해파리가 있는가?
- 색깔로 구분할 수 있는 해파리가 있는가?
- 하나의 사진에 해파리가 여러마리 들어있는 경우가 있는가?

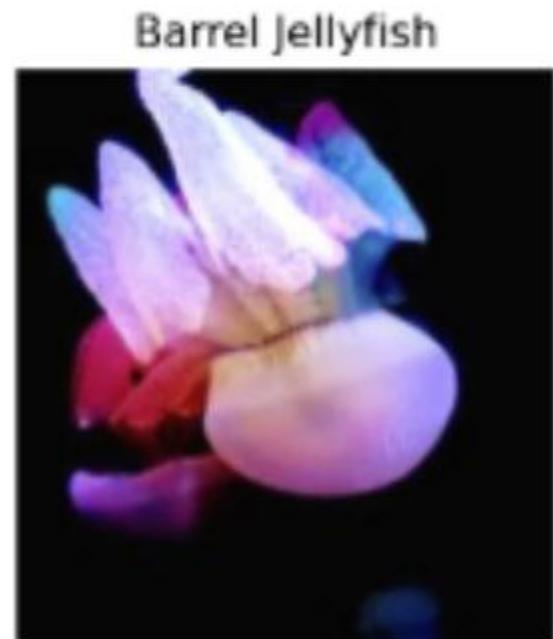
과정

- 클래스 별 TRAIN 데이터 20장씩 출력 후 육안 확인

2. EDA

EDA - 샘플 이미지 확인

2.2 샘플 이미지 확인

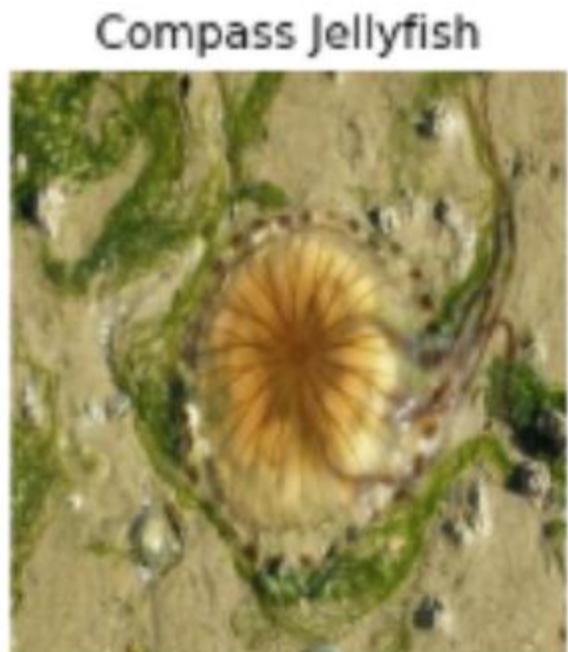
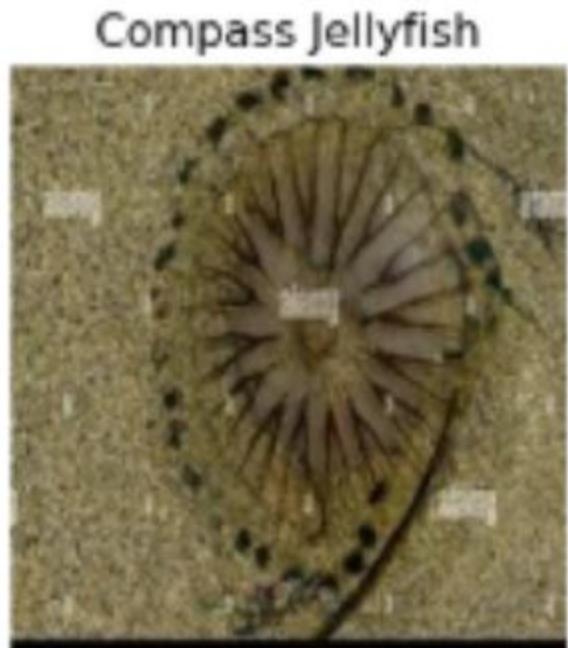


CAUSE & EFFECT

2. EDA

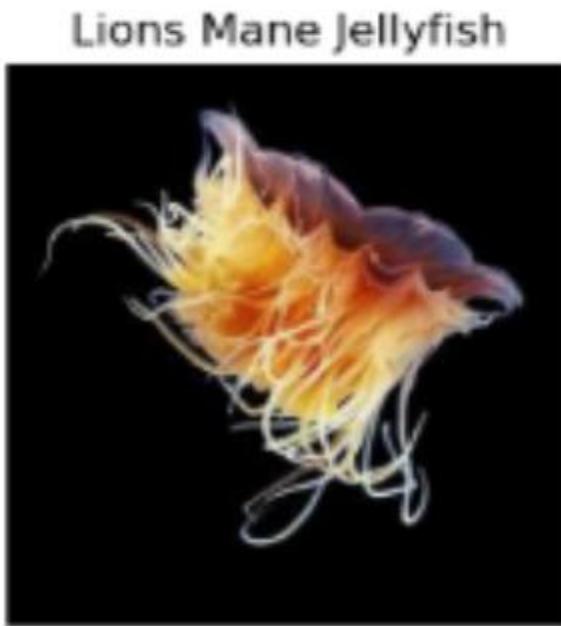
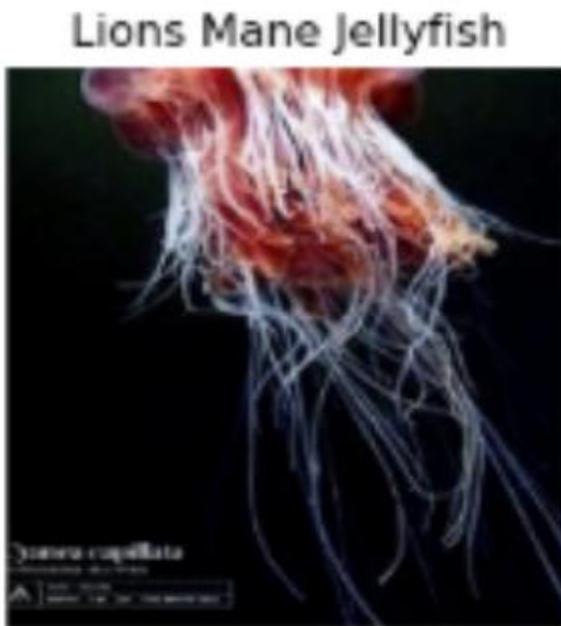
EDA - 샘플 이미지 확인

2.2 샘플 이미지 확인



2. EDA

EDA - 샘플 이미지 확인

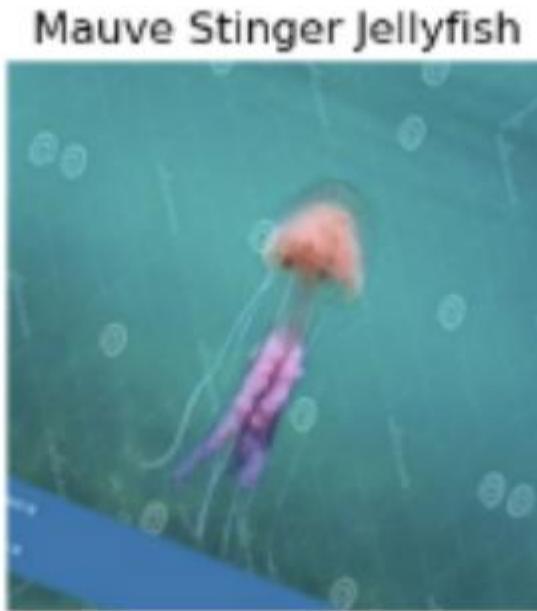


2.2 샘플 이미지 확인

2. EDA

EDA - 샘플 이미지 확인

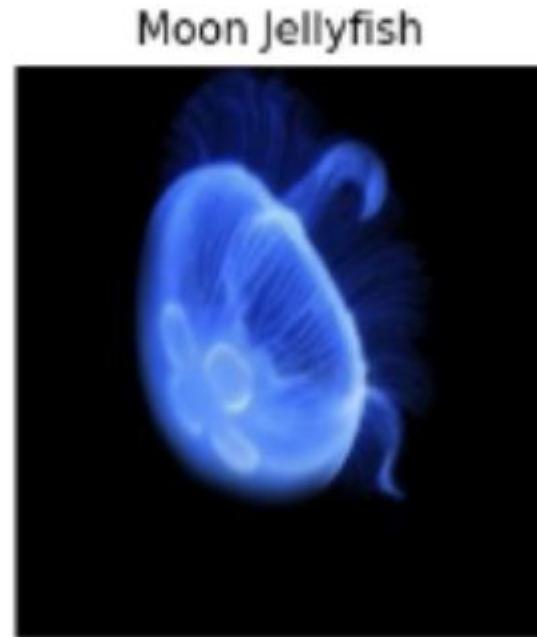
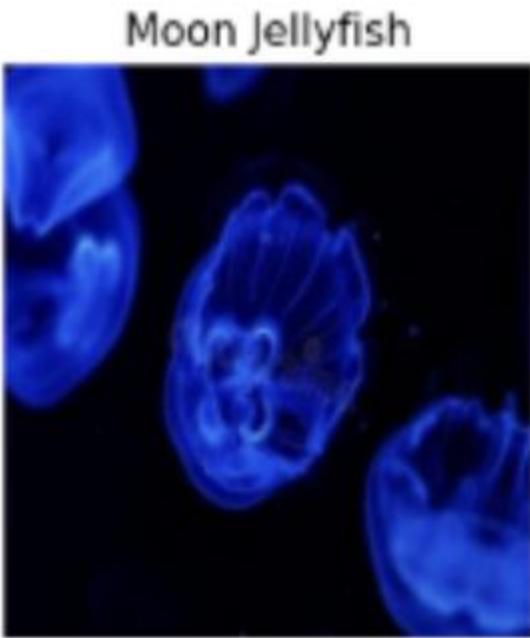
2.2 샘플 이미지 확인



2. EDA

EDA - 샘플 이미지 확인

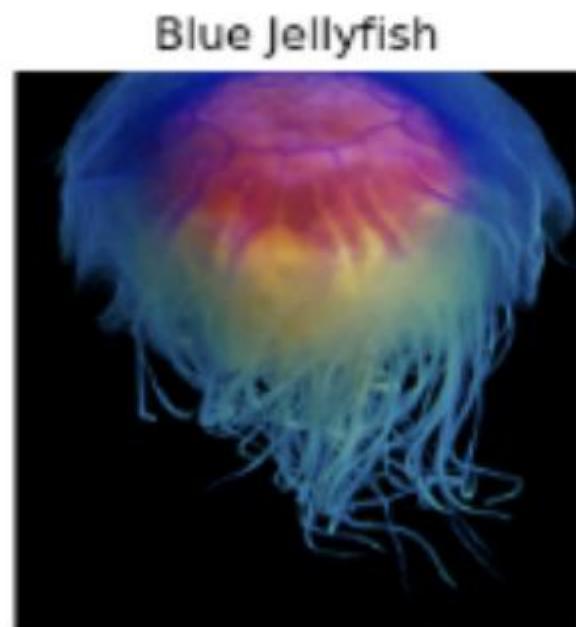
2.2 샘플 이미지 확인



2. EDA

EDA - 샘플 이미지 확인

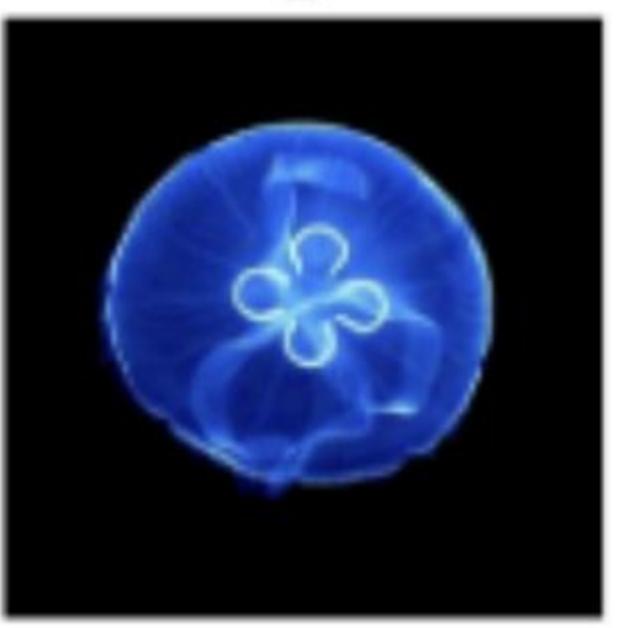
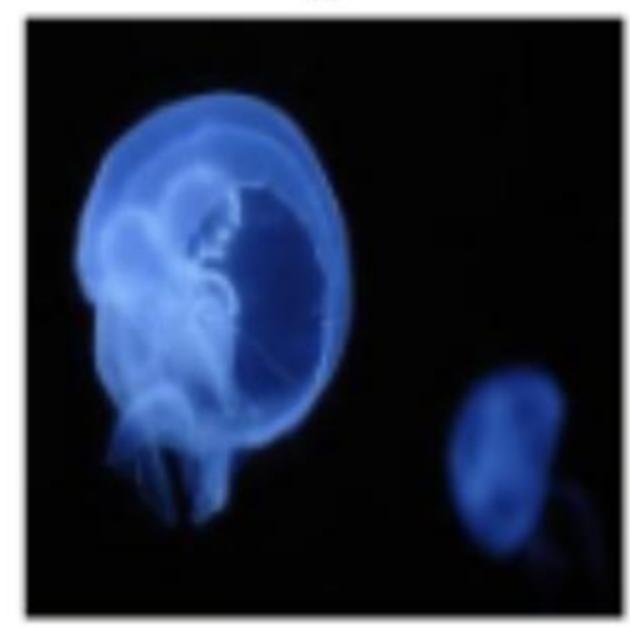
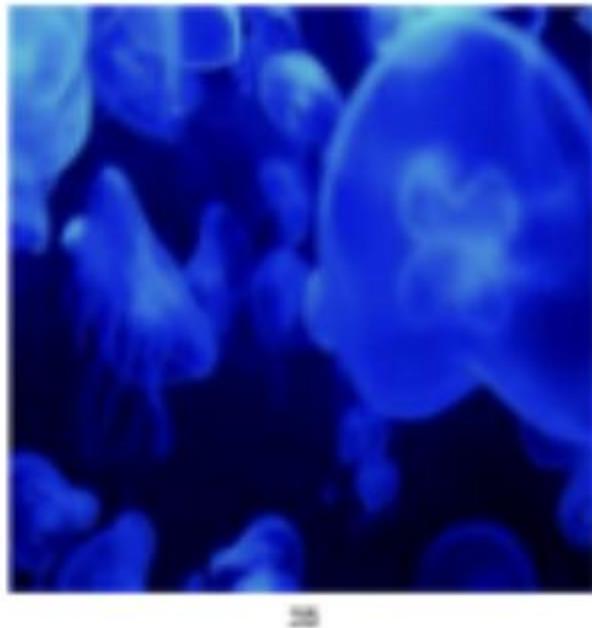
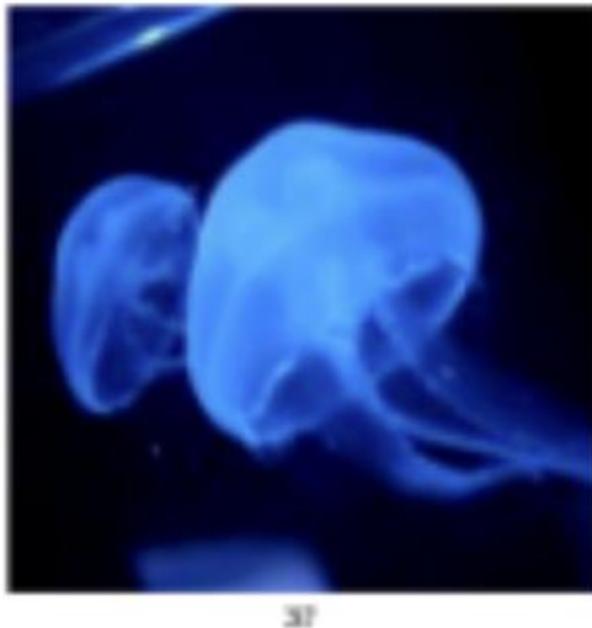
2.2 샘플 이미지 확인



2. EDA

EDA - 샘플 이미지 확인

2.2 샘플 이미지 확인



blue jellyfish

moon jellyfish

2. EDA

1 해파리 외 판별에 영향을 미칠 다른 요소들이 있는가?

- 결론: 배경색이 가지각색이라 판별에 영향을 미칠 수 있을 것으로 판단됨
- 대응 전략: GRAD-CAM을 통해서 판별이 가능할 것이나,
MOON과 BLUE JELLYFISH 를 제외한 모든 해파리는 고유의 색을 가지고 있기 때문에
다른 EDA를 살펴본 뒤 전략 수립 예정

2 특정 무늬나 형태로 구분할 수 있는 해파리가 있는가?

- 결론: 모든 해파리는 고유한 형태와 패턴을 가지고 있어 판별에 사용할 수 있음
- 대응 전략: 텍스처 시각화로 정확도를 높인다.

2.2 샘플 이미지 확인

2.

EDA

3 색깔로 구분할 수 있는 해파리가 있는가?

- 결론: 모든 해파리는 고유의 색깔을 가지고 있으나,
이미지의 빛 정도 등에 따라 색상의 왜곡이 있어 **판별에 사용하기는 어려움**
- 대응 전략: X

4 하나의 사진에 해파리가 여러마리 들어있는 경우가 있는가?

- 결론: 있음
- 대응 전략: 하지만 이것이 **판별에 영향을 미치는지는 따져볼 필요가 있음.**

2.2 샘플 이미지 확인

BACKGROUND

2.

EDA

2.2 샘플 이미지 확인

5

샘플 중 이미 증강 처리 된 데이터가 있음을 확인

- 대응 전략: 증강 및 미증강 이미지 수를 확인하고, 데이터 수를 본 후 판단

6

BLUE_JELLYFISH 샘플 속
MOON_JELLYFISH로 보이는 샘플 다수 발견

- 대응 전략:
 - 1) MOON JELLYFISH를 먼저 학습
 - 2) BLUE JELLYFISH 샘플 중 MOON JELLYFISH로 판별되는 이미지 샘플을 추출하여 제거
 - 3) 남은 BLUE JELLYFISH 이미지 셋만을 가지고 학습 (추가 증강 필요)

2. EDA

EDA - 유사도 확인 (색상)

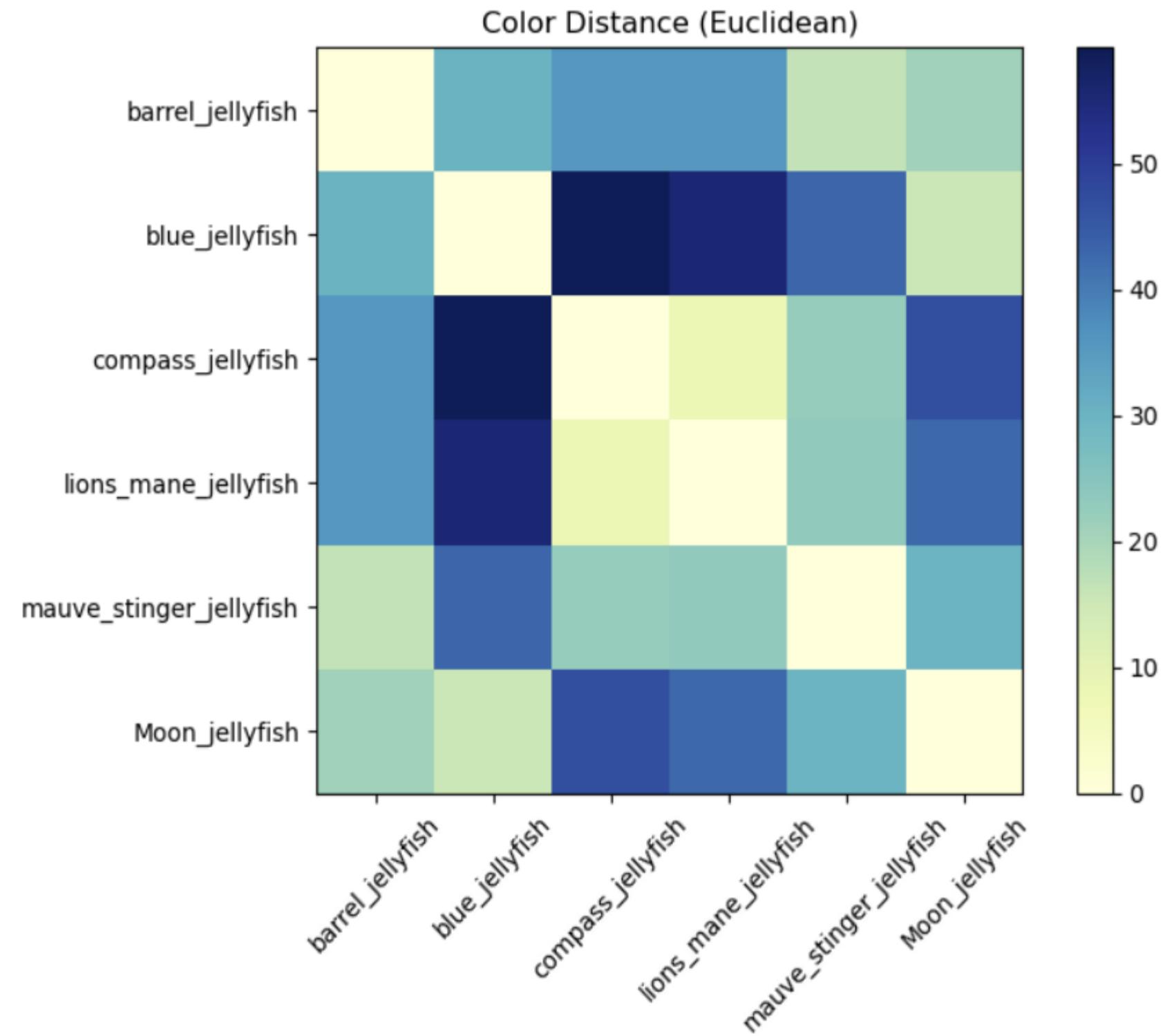
과정

- 각 이미지의 배경 제거
- 클래스 별 3가지 dominant color 추출
- 클래스 간 평균 대표 색상끼리 거리(Euclidean) 계산
- 유사도 시각화 & 분석

2.3 이미지 유사도(색상)

2. EDA

EDA - 유사도 확인 (색상)



2.3 이미지 유사도(색상)

2. EDA

EDA - 유사도 확인 (색상)

- 대표 색상 유사도가 가장 가까운 클래스 TOP 5 (Euclidean 거리):
 compass_jellyfish ↔ lions_mane_jellyfish : 거리 = 8.44
 blue_jellyfish ↔ Moon_jellyfish : 거리 = 15.29
 barrel_jellyfish ↔ mauve_stinger_jellyfish : 거리 = 16.51
 barrel_jellyfish ↔ Moon_jellyfish : 거리 = 20.95
 compass_jellyfish ↔ mauve_stinger_jellyfish : 거리 = 22.58

2.3 이미지 유사도(색상)

2.

EDA

1

COMPASS <-> LIONS_MANE (거리 8.44): 매우 유사함.

2

유사도 15~25 사이의 유사한 색상 그룹 다수

- 대응 전략: 모델이 형태 또는 텍스처에 집중하도록 해야할지 판단 필요

2.3 이미지 유사도(색상)

BACKGROUND

— 2. EDA

EDA - 유사도 확인 (형태)

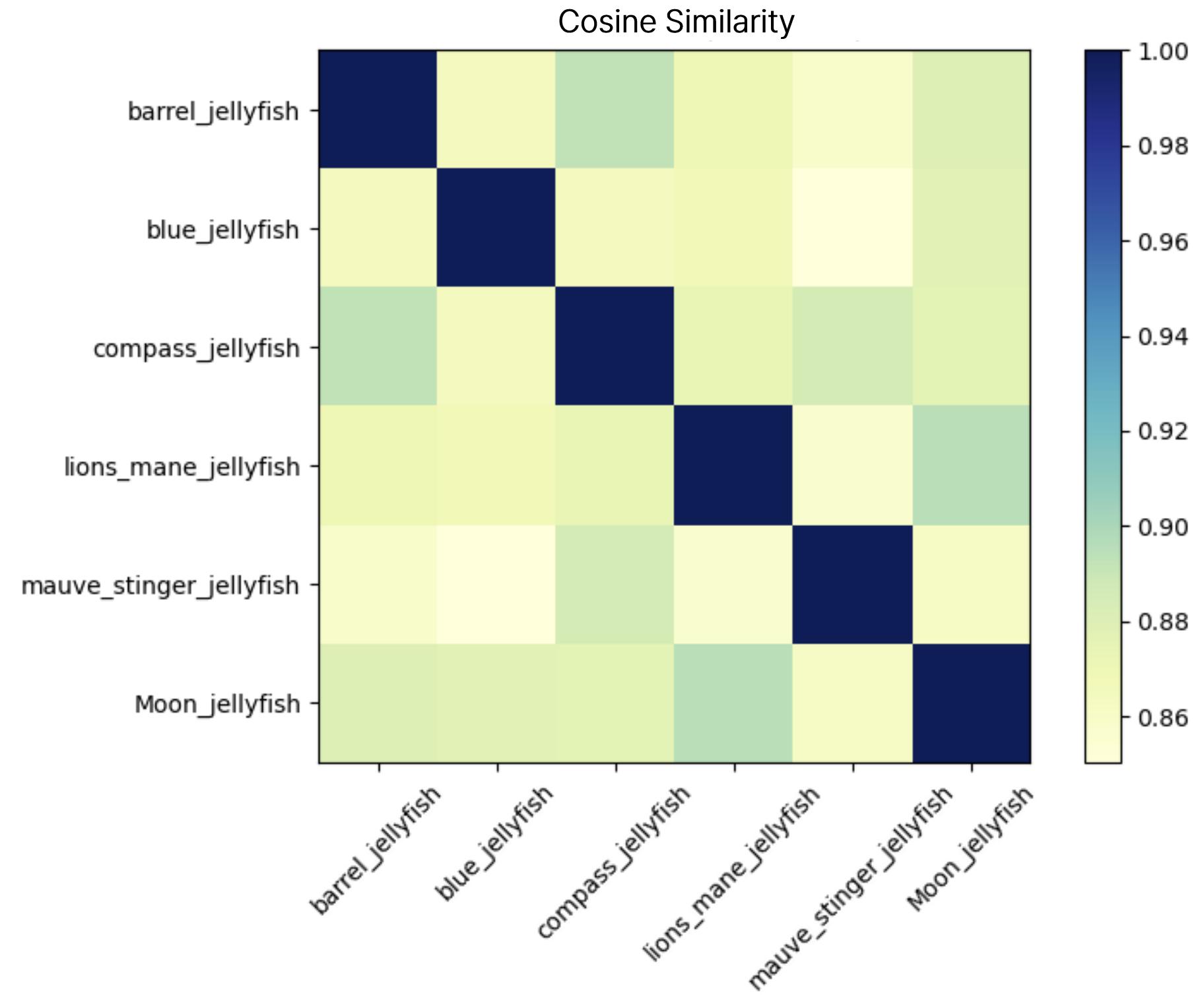
과정

- 각 이미지의 배경 제거
- 이미지 resize & 흑백으로 변환(grayscale)
- HOG descriptor 추출
- 클래스 당 여러 이미지 -> 평균 HOG 벡터 생성
- 클래스 간 cosine 유사도 계산
- 유사도 시각화 & 분석

2.4 이미지 유사도(형태)

2. EDA

EDA - 유사도 확인 (형태)



2.4 이미지 유사도(형태)

2.

EDA

EDA - 유사도 확인 (형태)

▣ 형태 기반 유사도가 높은 클래스 TOP 5:

`lions_mane_jellyfish ↔ Moon_jellyfish` : 유사도 = 0.8953

`barrel_jellyfish ↔ compass_jellyfish` : 유사도 = 0.8925

`compass_jellyfish ↔ mauve_stinger_jellyfish` : 유사도 = 0.8848

`barrel_jellyfish ↔ Moon_jellyfish` : 유사도 = 0.8807

`blue_jellyfish ↔ Moon_jellyfish` : 유사도 = 0.8777

2. EDA

1

Moon_jellyfish가 형태 유사도 top 5개 중
3개의 쌍에서 나타남

- 대응 전략: 분리 학습 또는 샘플 강조 등 별도 처리 필요

▣ 형태 기반 유사도가 높은 클래스 TOP 5:

lions_mane_jellyfish ↔ Moon_jellyfish : 유사도 = 0.8953

barrel_jellyfish ↔ compass_jellyfish : 유사도 = 0.8925

compass_jellyfish ↔ mauve_stinger_jellyfish : 유사도 = 0.8848

barrel_jellyfish ↔ Moon_jellyfish : 유사도 = 0.8807

blue_jellyfish ↔ Moon_jellyfish : 유사도 = 0.8777

— 2. EDA

EDA - 유사도 확인 (텍스처)

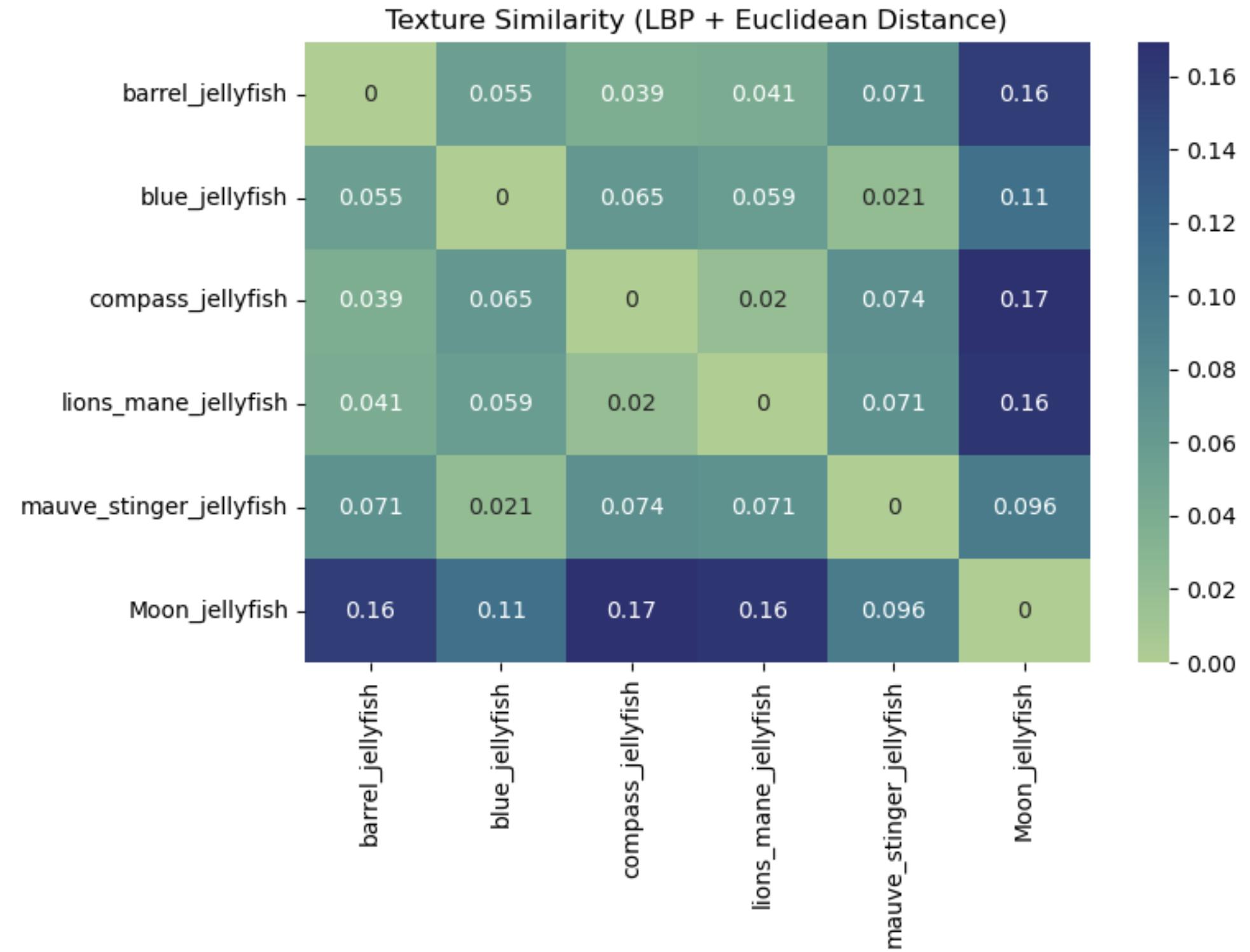
과정

- LBP (Local Binary Pattern) 히스토그램 추출
- 클래스 별 평균 LBP 히스토그램 생성
- 클래스 간 유사도 계산 (Euclidean Distance)
- 유사도 시각화 & 분석

2.5 이미지 유사도(텍스처)

2. EDA

EDA - 유사도 확인 (텍스처)



2.5 이미지 유사도(텍스처)

2. EDA

1

유클리디안 거리 값이 전반적으로 매우 낮은 수준 (0.02~0.17)

- 대응 전략: 해파리의 실제 텍스처 패턴(무늬나 선명도)은 배경에 의해 희석되거나 가려질 수 있으므로,
배경 제거가 필요할 것임

2

다른 클래스 대비 Moon_jellyfish는 확연히 큰 거리를 보임

- 대응 전략: Blue_jellyfish 내에 Moon_jellyfish 샘플이 섞여있어
모델이 이 두 종을 텍스처 특성 기반으로 혼동할 수 있음.
따라서 **Blue_jellyfish** 내에서 **Moon_jellyfish**를 속아내야 함.

2.5 이미지 유사도(텍스처)

2. EDA

- 배경 제거
- Blue 내에서 Moon 숨아내기



이미지 색상 모두 유사
(euclidean distance 50 이하)

이미지 유사도 결론



이미지 형태 모두 유사
(모두 85% 이상)



이미지 텍스처 전반적으로
유사하나,
Moon_jellyfish 뿐.

2. EDA

EDA – 클래스 별 이미지 수 확인

Why?

클래스 간 데이터 imbalance 확인

과정

- LBP(Local Binary Pattern) 히스토그램 추출
- 클래스 별 평균 LBP 히스토그램 생성
- 클래스 간 유사도 계산 (Euclidean Distance)
- 유사도 시각화 & 분석

2.5 클래스 별 이미지 수 확인

2. EDA

2.5 클래스 별 이미지 수 확인

EDA – 클래스 별 이미지 수 확인

- 발견사항:

- 약한 불균형은 존재하지만, 문제 될 정도는 아님.
- 가장 원본이 적은 클래스는 41장 (27%)
- 가장 원본이 많은 클래스는 61장 (40%)

- 대응 전략:

- 큰 문제 없으나,
필요 시 early stopping, val set 기준 평가,
mixup 등으로 커버 가능

	Class	Total	Original	Augmented
0	barrel_jellyfish	150	41	109
1	blue_jellyfish	150	55	95
2	compass_jellyfish	150	57	93
3	lions_mane_jellyfish	150	61	89
4	mauve_stinger_jellyfish	150	58	92
5	Moon_jellyfish	150	47	103



2. EDA

EDA – 중복 이미지 탐지



동일한 데이터를 여러 번 학습시키면 과적합 우려 증가



- 완전 동일한 이미지 탐지
 - 바이트 해시 (MD5)를 통해 이미지 내용을 바이트 읽고
 - hash를 계산해서 비교
- 유사 이미지 탐지
 - imagehash 라이브러리를 사용하여 비슷한 이미지 탐지

2. EDA

EDA – 중복 이미지 탐지

- 발견 내용:
 - 완전 중복 이미지 x
 - 유사한 이미지 o(66쌍)
 - 약한 데이터 증강으로 추측
- 대응 전략:
 - 유사한 이미지가 학습에 유용할지 방해될지 판단 필요
 - **추후 필요하다면, 약한 증강 이미지 삭제 후 샘플 수를 맞춰 재증강**

2.6 중복 이미지 탐지

similar images 23



Image 1

Image 2



similar images 24



Image 1

Image 2



— 2. EDA

EDA – 이미지 품질 확인



저품질의 데이터를 학습시키면 모델 품질 또한 저하



- 흐린 / 어두운 이미지의 개수 체크
- 흐린 / 어두운 이미지 시각화

2. EDA

EDA – 이미지 품질 확인

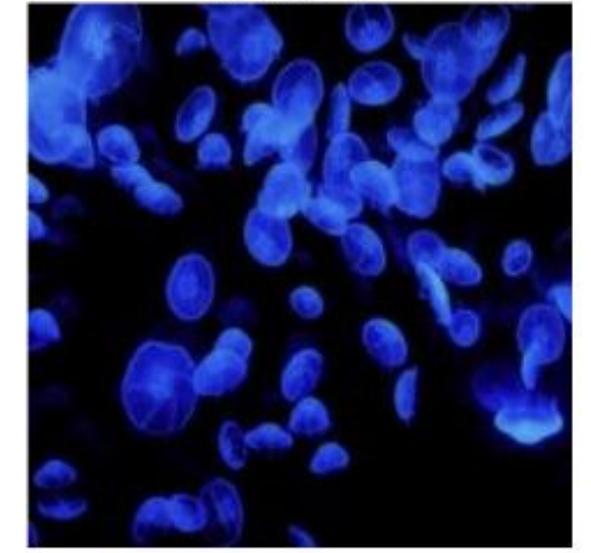
- 발견 내용:
 - 저품질 이미지 : 10장 (/900장)
 - 어두운 이미지: 약 25장
 - 모두 배경이 어두운 경우인 것으로 확인
- 대응 전략:
 - 흐린 이미지의 경우 수가 많지 않으므로 대응 X
 - 어두운 이미지의 경우 배경만 어두우며 해파리는 밝으므로 대응 X

2.7 이미지 품질 확인

blue_jellyfish
04.jpg
Bright: 22.1



blue_jellyfish
07.jpg
Bright: 24.5



blue_jellyfish
45.jpg
Bright: 22.6



blue_jellyfish
54.jpg
Bright: 17.5



blue_jellyfish
62.jpg
Bright: 21.5

lions_mane_jellyfish
55.jpg
Bright: 23.5

2.

EDA

EDA – 결론 (전처리 전략)

- 1 (확인 후 반영) 배경색 제거
- 2 Blue 해파리 내 Moon 해파리 속아내기
- 3 (확인 후 반영) 한 이미지 내의 복수 해파리가 있는 경우 제거
- 4 (사후 대응) 학습 데이터셋 부족한 경우 증강
- 5 (사후 대응) 66쌍의 유사한 이미지 쌍에 대해,
추후 필요 시 유사 이미지 제거 후 재증강

—

3. 데이터 전처리

데이터 전처리 – 배경 제거



3.1 배경 제거

3. 데이터 전처리

데이터 전처리 – Blue <-> Moon 속아내기

1 모델 선택 전략

항목	선택 사항	선택 이유
모델	MobileNetV2 (imagenet pretrained)	- 경량 모델이라 학습 빠르고 - 작은 데이터셋에서도 과적합 덜함 - 이미지 분류에 널리 검증됨
구조	head만 추가 GlobalPool -> Dense(64) -> Dense(1, sigmoid)	표현력 + 경량성 밸런스

2 분류 전략

항목	설명
0~1 출력	sigmoid 결과값 -> moon일 확률
판별 기준값 (threshold)	기본 0.5 사용 (moon일 확률 > 50%)
추가 검증	이미지를 뽑아서 육안으로 확인 후, threshold 조정

3.2 blue – moon 속아내기

3. 데이터 전처리

데이터 전처리 – Blue <-> Moon 속아내기

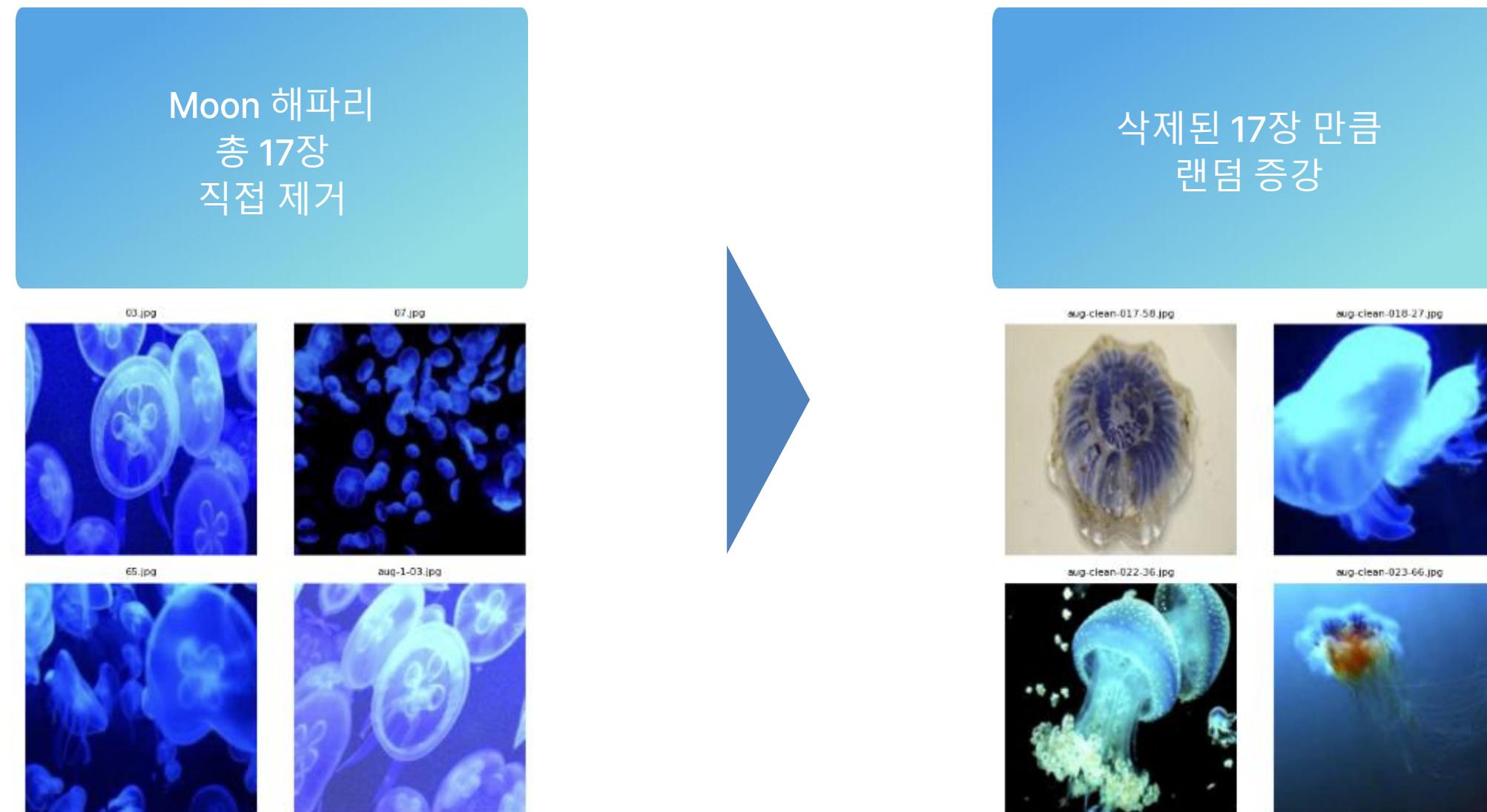
- 발견 사항:
 - moon jellyfish 일 확률 상위 10개 이미지를 보았을 때 대체적으로 잘 골라냄
 - 하지만 일부 blue jellyfish로 추정되는 것이 섞여있음
 - 또한, moon 추정 확률 또한 **1위가 14%**로 매우 낮음
- 대응 전략:
 - 모든 이미지를 눈으로 확인 후 수작업으로 골라냄
 - 보수적인 판별을 위해,
우산 내부에 말굽 모양 생식선 존재할 경우만 선택



3.2 blue – moon 속아내기

3. 데이터 전처리

데이터 전처리 – Blue <-> Moon 속아내기



3.2 blue – moon 속아내기

3. 데이터 전처리

데이터 전처리 – 한 이미지 내 복수 해파리

목표

하나의 학습 이미지 내에 해파리가 여러마리 존재할 경우
모델의 성능에 영향을 미치는지 확인

과정

- 두 가지 케이스에 대하여 모델 성능 대조 분석
 - 단일 개체만 존재하는 학습 이미지 데이터셋
 - 복수 개체가 포함된 학습 이미지 데이터셋 (기존 데이터셋)
- Grad-CAM을 통해 모델 학습 내용 시각화

3. 데이터 전처리

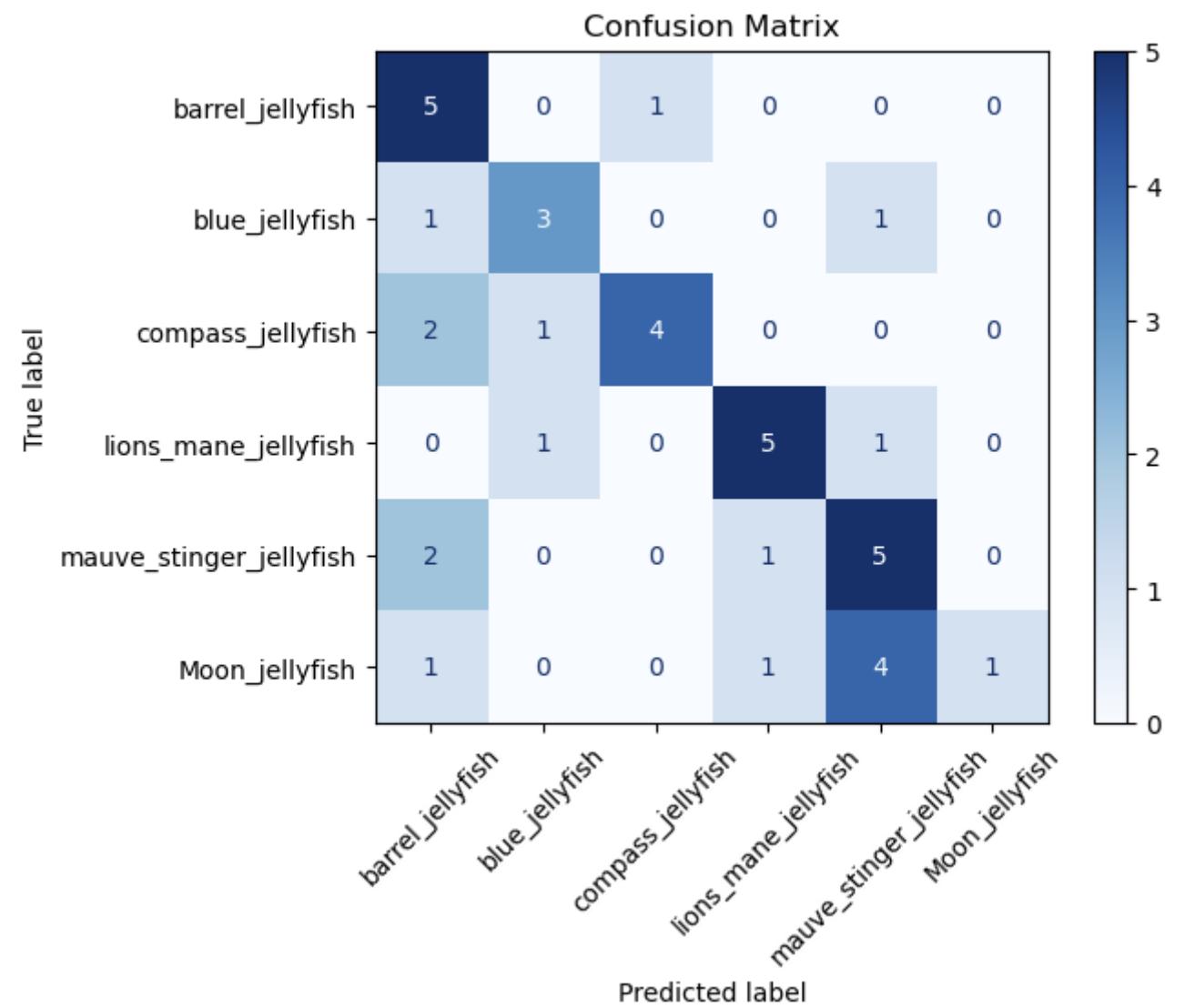
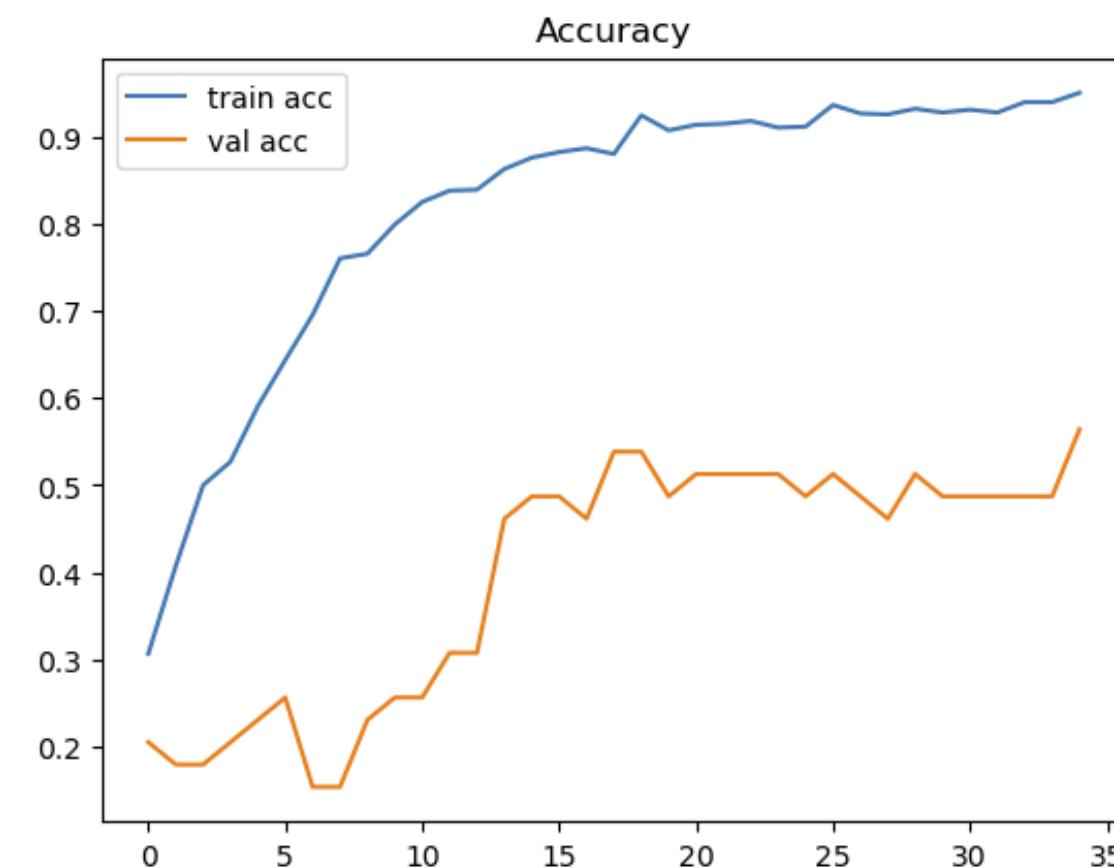
데이터 전처리 – 한 이미지 내 복수 해파리

- 샘플 제거 기준:
 - 비슷한 크기의 객체가 복수 존재 시 무조건 제거
 - 가장자리에 애매하게 다른 객체가 보일 경우는 보류
- 샘플 확인 방법:
 - 육안 확인

- ✓ barrel_jellyfish: 유지 134장 / 제거 12장
- ✓ mauve_stinger_jellyfish: 유지 145장 / 제거 5장
- ✓ compass_jellyfish: 유지 139장 / 제거 5장
- ✓ blue_jellyfish: 유지 131장 / 제거 17장
- ✓ Moon_jellyfish: 유지 111장 / 제거 39장
- ✓ lions_mane_jellyfish: 유지 139장 / 제거 5장

3. 데이터 전처리

데이터 전처리 – 한 이미지 내 복수 해파리



3.3 복수 개체 대응

3. 데이터 전처리

데이터 전처리 – 한 이미지 내 복수 해파리

전반적 지표

항목	제거 전	제거 후	분석
Train Accuracy	0.95 이상	0.92 이상	큰 차이 없음, 둘 다 충분히 학습
Val Accuracy	최대 0.58	최대 0.58	변화 거의 없음
Test Accuracy	0.475	0.50	미세한 상승, 하지만 큰 차이는 아님
Test Loss	1.77	1.76	loss는 거의 비슷함
Macro F1	0.46	0.50	소폭 개선됨 (class 간 균형 향상)
Confusion Matrix	blue ↔ moon 혼동 / compass 오 예측	blue ↔ moon 여전히 혼동 / compass 개선	일부 클래스 분별력 향상, 특히 compass_jellyfish

3.3 복수 개체 대응

3. 데이터 전처리

데이터 전처리 – 한 이미지 내 복수 해파리

클래스	f1-score 변화	해석
barrel_jellyfish	0.55 → 0.60	비슷
blue_jellyfish	0.53 → 0.44	감소
compass_jellyfish	0.22 → 0.40	크게 개선
lions_mane_jellyfish	0.55 → 0.46	감소
mauve_stinger_jellyfish	0.43 → 0.62	크게 개선

3.3 복수 개체 대응

3. 데이터 전처리

데이터 전처리 – 한 이미지 내 복수 해파리

[결론]

- 복수 개체 이미지 제거가 클래스 별로 보이는 효과가 다름.
 - blue_jellyfish, lions_mane_jellyfish의 f1는 감소했으나,
 - compass, mauve_stinger는 분명한 향상
- 이는 모델이 해파리 외의 요소(배경 등)를 참고했을 가능성 있음을 시사.
 - Grad-Cam 시각화 필요

4. 베이스라인 모델 설계

4.1 모델 투아보기

베이스라인 모델 설계 – 모델 투아보기

[베이스라인 모델 요구사항]

1. 데이터 특성과 모델 구조 간의 적합성

Jellyfish 데이터셋은 클래스 간 색상과 형태의 유사성이 높고,
데이터 수가 제한적이기 때문에,
지나치게 깊거나 복잡한 모델은 과적합 우려가 있음.

-> 적당한 표현력과 경량성의 균형이 맞는 모델 필요

2. 모델 경량성과 실험 효율성

베이스라인이므로, 빠른 실험 사이클을 가능하게 하며
적절한 수준의 성능까지 보장하는 모델 필요

4. 베이스라인 모델 설계

베이스라인 모델 설계 – 모델 틉아보기

- 1 CNN
- 2 MobilenetV2
- 3 MobilenetV3
- 4 ConvNext
- 5 Resnet50

4.1 모델 틉아보기

4. 베이스라인 모델 설계

베이스라인 모델 설계 – 모델 투아보기

- 1 CNN
 - 2 MobilenetV2
 - 3 MobilenetV3
 - 4 ConvNext
 - 5 Resnet50
- 표현력 good
- 경량성 good

4.1 모델 투아보기

4. 베이스라인 모델 설계

베이스라인 모델 설계 – 모델 투아보기

핵심 아웃풋 지표

지표	설명	베이스라인 선택에 미치는 영향
Confusion Matrix	어떤 클래스에서 자주 틀리는지	형태 vs 색상 헷갈림 여부 분석
GradCAM	모델이 어디를 보고 분류했는가	모델이 의미 있는 영역 보는지 확인
학습 곡선	Overfitting or underfitting 빠르게 확인	복잡도에 비해 충분한 학습이 되는지

4.1 모델 투아보기

4. 베이스라인 모델 설계

베이스라인 모델 설계 – 모델 설계

Layer (type)	Output Shape	Param #
MobileNetV3Large (Functional)	(None, 7, 7, 960)	2,996,352
global_average_pooling2d_1 (GlobalAveragePooling2D)	(None, 960)	0
batch_normalization_1 (BatchNormalization)	(None, 960)	3,840
dense_1 (Dense)	(None, 256)	246,016
dropout_1 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 6)	1,542

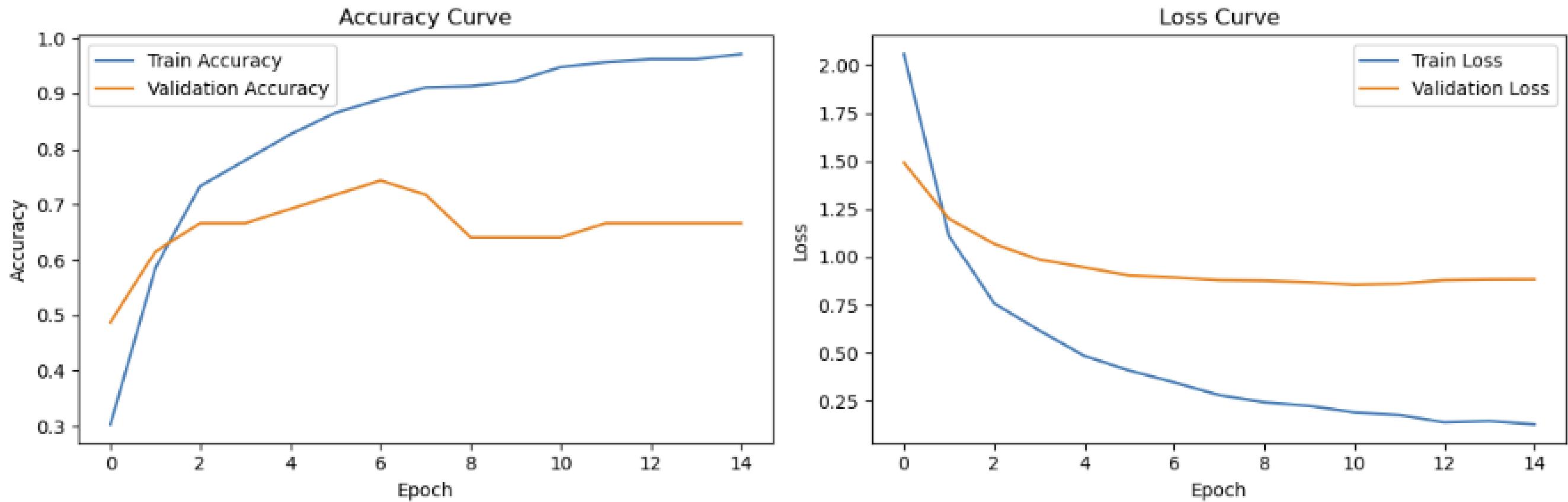
Total params: 3,247,750 (12.39 MB)

Trainable params: 405,318 (1.55 MB)

Non-trainable params: 2,842,432 (10.84 MB)

4. 베이스라인 모델 설계

베이스라인 모델 설계 – 모델 설계



테스트 정확도: 0.9250 (매우 우수)

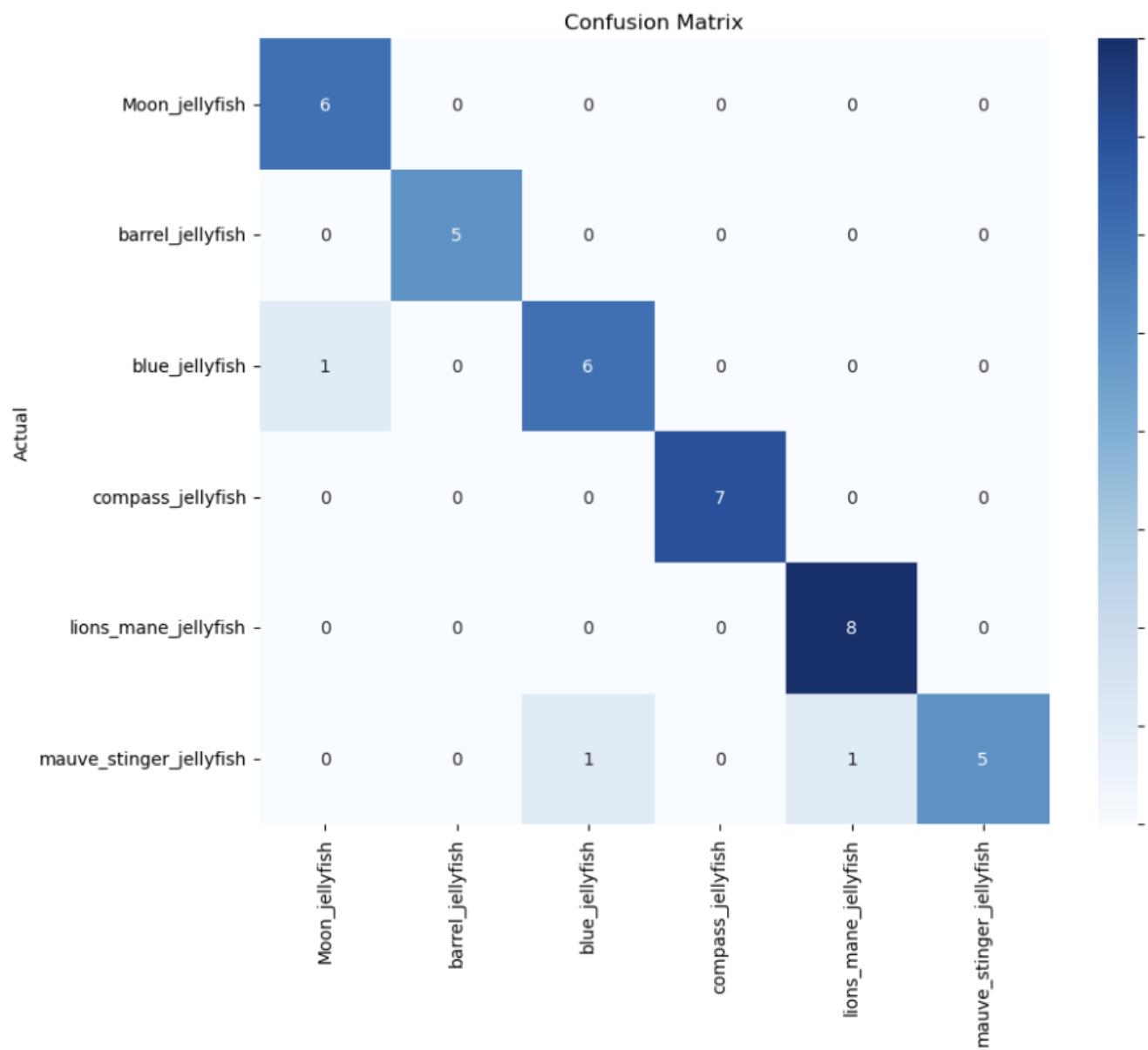
테스트 손실: 0.3479

4.2 모델 설계

학습 곡선: 학습은 우수, 검증은 약간 overfitting 조짐 (6~7 에포크 부근에서 val_acc 감소)

4. 베이스라인 모델 설계

베이스라인 모델 설계 – 모델 설계

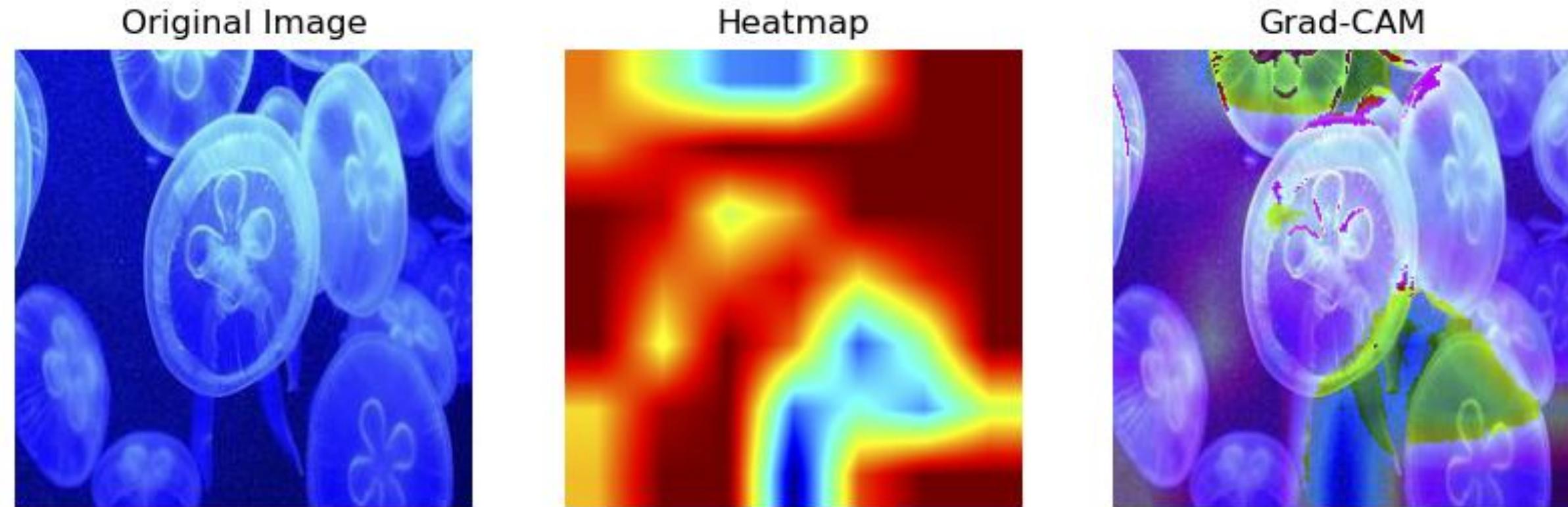


4.2 모델 설계

대부분 클래스 정확하게 분류, 일부 혼동 (mauve : blue)

4. 베이스라인 모델 설계

베이스라인 모델 설계 – 모델 설계



상단 가장자리 해파리는 잘 잡았으나, 그 외에는 다소 부족한 feature 학습 확인

4.2 모델 설계

4. 베이스라인 모델 설계

베이스라인 모델 설계 – 전처리 데이터 성능 검증

1.

배경 제거한 데이터셋과 그렇지 않은 데이터셋의 성능 비교

2.

중복 개체가 포함된 데이터를 제거한 셋과 그렇지 않은 셋의 비교

4. 베이스라인 모델 설계

4.3 전처리 데이터 성능 검증

베이스라인 모델 설계 – 전처리 데이터 성능 검증

배경 제거

지표	배경 제거 x	배경 제거 o
테스트 손실	0.3479	0.2614
테스트 정확도	0.9250	0.9556
학습 곡선	학습은 우수, 검증은 약간 overfitting 조짐 (6~7 에포크 부근에서 정확도 감소)	안정적인 우상향
Confusion Matrix	대부분 클래스 정확하게 분류, 일부 혼동 (mauve ↔ blue)	대부분 클래스 정확하게 분류
Grad_Cam	상단 가장자리 해파리는 잘 잡았으나, 그 외에는 다소 부족한 feature 학습 확인	중앙부 해파리만 남아, 정확한 feature 학습 확인

결론: 배경 제거는 적용

- 테스트 정확도 상승 있었으며,
- 학습 곡선 상으로 아직 과적합 양상 또한 보이지 않음.
- 따라서, 에포크 증가 시 전처리 전 대비 나은 성능을 보일 것으로 예상됨.

4. 베이스라인 모델 설계

4.3 전처리 데이터 성능 검증

베이스라인 모델 설계 – 전처리 데이터 성능 검증

중복 개체 제거

지표	중복 개체 제거 x	중복 개체 제거 o
테스트 손실	0.3479	0.3271
테스트 정확도	0.9250	0.9000
학습 곡선	학습은 우수, 검증은 약간 overfitting 조짐 (6~7 에포크 부근에서 정확도 감소)	비교적 꾸준한 우상향
Confusion Matrix	대부분 클래스 정확하게 분류, 일부 혼동 (mauve ↔ blue)	비교적 다양한 클래스에서 혼동
Grad_Cam	-	해파리 중심부에 올바른 feature 학습 확인

결론: 중복 개체 제거 전처리 성능은 좋으나 미적용

- 테스트 정확도의 소폭 하락 있었으나,
학습 데이터 수 감소(900-> 800) 으로 인한 영향까지 생각하면 매우 미미한 수준.
- 학습 곡선 상에서도 뚜렷한 과적합 양상 보이지 않아, 에포크 늘림으로써 정확도 향상 가능성 있어보임.
- 다만, 해당 전처리가 들어간다면, test 셋에서도 동일하게 중복 개체 제거하는 전처리가 필요함.
- 이는 불가능한 작업이므로, 제외시킴.

—

4.

베이스라인 모델 설계

베이스라인 모델 설계 – 전처리 데이터 성능 검증

최종 전처리 적용 사항

- 1 배경 제거
- 2 Blue 내 Moon 속아내기

4.3 전처리 데이터 성능 검증

5. 모델 개선

모델 개선 – 여러 모델 테스트

1. CNN

- 특징: 전통적인 Conv → ReLU → Pool 구조 기반의 단순한 CNN
- 장점: 구조 이해가 쉬움, 빠른 실험 가능
- 단점: 성능 한계 명확

2. mobilenet v2

- 특징: 경량화 모델 (모바일용)
- 장점: 빠름, 극소량의 자원에서도 학습 가능
- 단점: 성능 ceiling 낮음

3. ConvNext

- 특징: 최신 CNN 계열 (ViT 스타일 구조 도입)
- 장점: 최신성, 성능 좋음
- 단점: 무거움, baseline으로 과할 수도

4. Resnet50

- 특징: skip connection으로 딥러닝 안정화
- 장점: 높은 성능, 빠른 수렴
- 단점: 구조 복잡도 다소 있음

5. 모델 개선

모델 개선 – CNN

베이스 모델과 비교 포인트

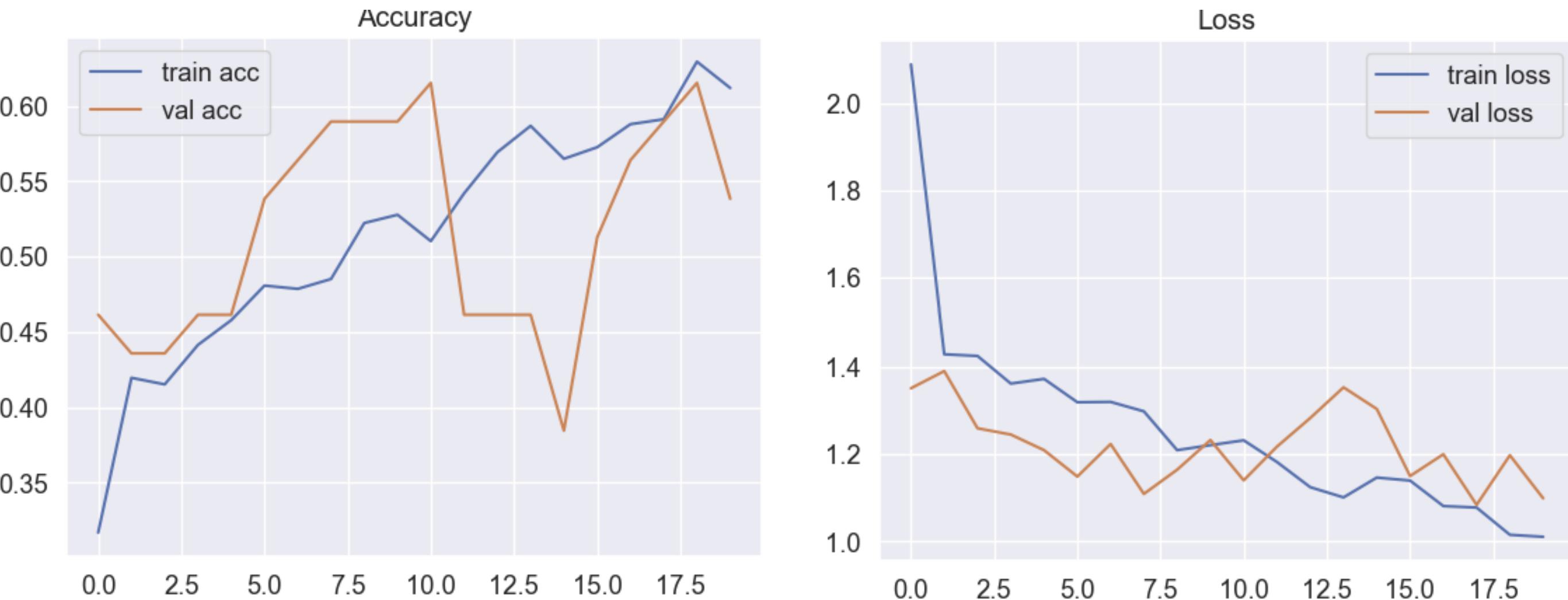
항목	설명
장점	파라미터 수 적고 빠름, 과적합 조절 쉬움
베이스라인 대비 기대 효과	- 유사 클래스 구분에 대한 형태 기반 커스터마이징 가능 - 작은 데이터셋에서는 오히려 더 성능 안정적

5.2 모델 개선 – CNN

5.

모델 개선

모델 개선 – CNN

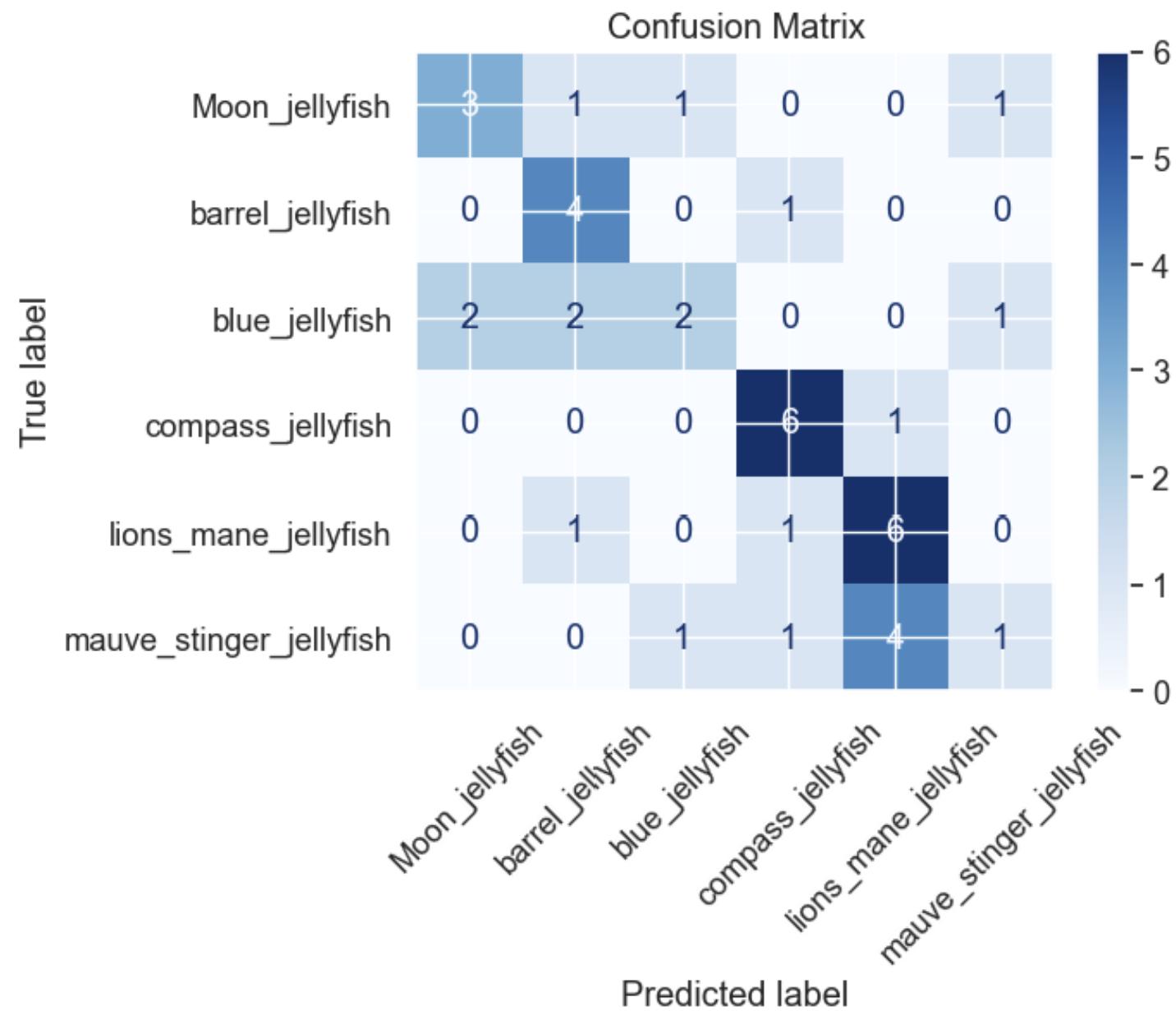


5.2 모델 개선 – CNN

5. 모델 개선

모델 개선 – CNN

5.2 모델 개선 – CNN



5. 모델 개선

모델 개선 – CNN

결과 확인

- 테스트 정확도: 0.5500, 테스트 손실: 1.0828 로, 베이스라인 모델과 비교 불가할 정도로 현저히 낮은 성능.

원인 예상

1. Feature Extractor의 표현력 차이

- MobileNetV3는 수십 개의 depthwise/separable conv를 가지고 있어 미세한 차이 감지 능력이 우수
- 하지만 CNN은 단순 Conv2D + MaxPooling 구조로, 복잡한 패턴을 놓치기 쉬움

2. 전처리 전략과의 상호작용

- 우리가 적용한 전처리 전략은 미세한 정보 차이를 더 부각시키는 작업임.
- 하지만 CNN은 그 미세한 차이를 포착할 수 있을 정도로 정교하지 못함.

5. 모델 개선

모델 개선 – MobilenetV2

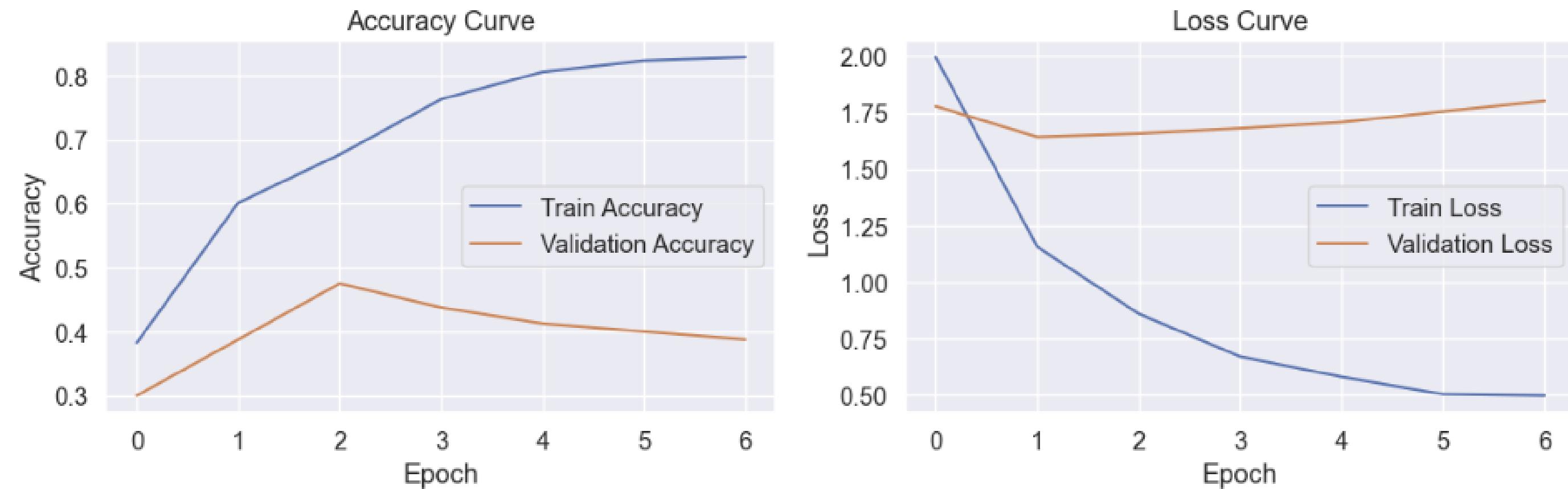
베이스 모델과 비교 포인트

항목	설명
장점	동일한 MobileNet 계열 구조로 비교 기준 동일하나, 더 빠르고 얇음
베이스라인 대비 기대 효과	- 구조가 더 간단해, 학습이 빠르고 일반화 성능이 좋을 수 있음 - 성능은 비슷하지만 속도 개선 가능

5.

모델 개선

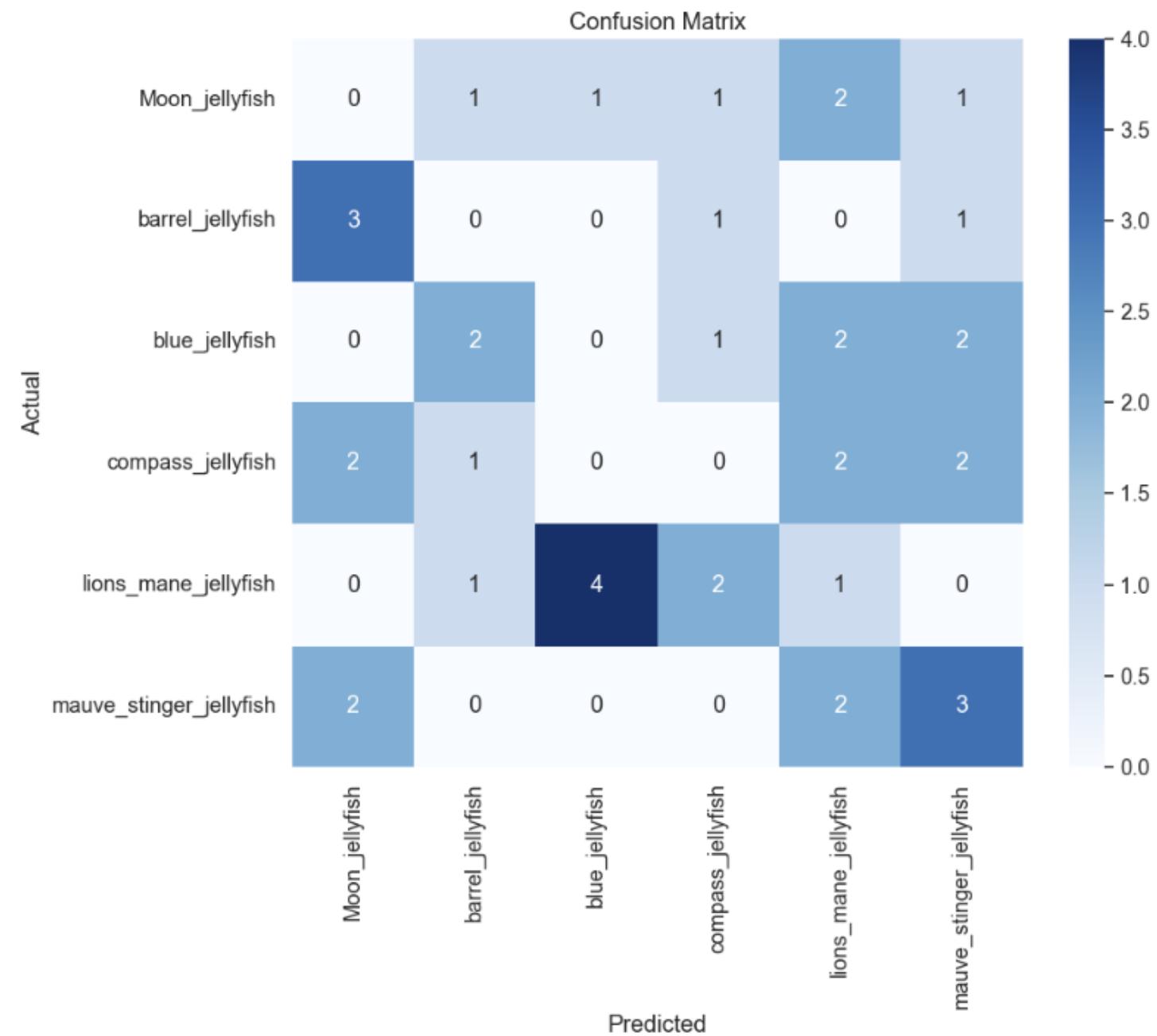
모델 개선 – MobilenetV2



5.3 모델 개선 – MobilenetV2

5. 모델 개선

모델 개선 – MobilenetV2



5.3 모델 개선 – MobilenetV2

5.

모델 개선

5.3 모델 개선 – MobilenetV2

모델 개선 – MobilenetV2

결과 확인

- 테스트 정확도: 0.7750, 테스트 손실: 0.5895로, 베이스라인 모델 대비 상당히 낮은 성능.
- 2 에포크부터 떨어지는 val accuracy 확인.

원인 예상

1. 이론 과적합

- V2는 기본적으로 얇아서 feature 추출력이 부족함.
- 하지만 우리 task 및 전처리 전략은 미세한 차이에 집중하도록 함.
- V2는 빠르게 학습데이터에 적응하고 일반화를 하지 못해서 train은 상승하나 val은 하락함

2. 데이터 특성 및 전처리 전략과의 상호작용

- 해파리 데이터는 미세한 형태/색/텍스처를 보고 판단해야하며,
- 우리가 적용한 전처리 전략은 미세한 정보 차이를 더 부각시키는 작업임.
- 하지만 V2는 그 미세한 차이를 포착할 수 있을 정도로 깊지 못함.

5. 모델 개선

모델 개선 – ConvNext

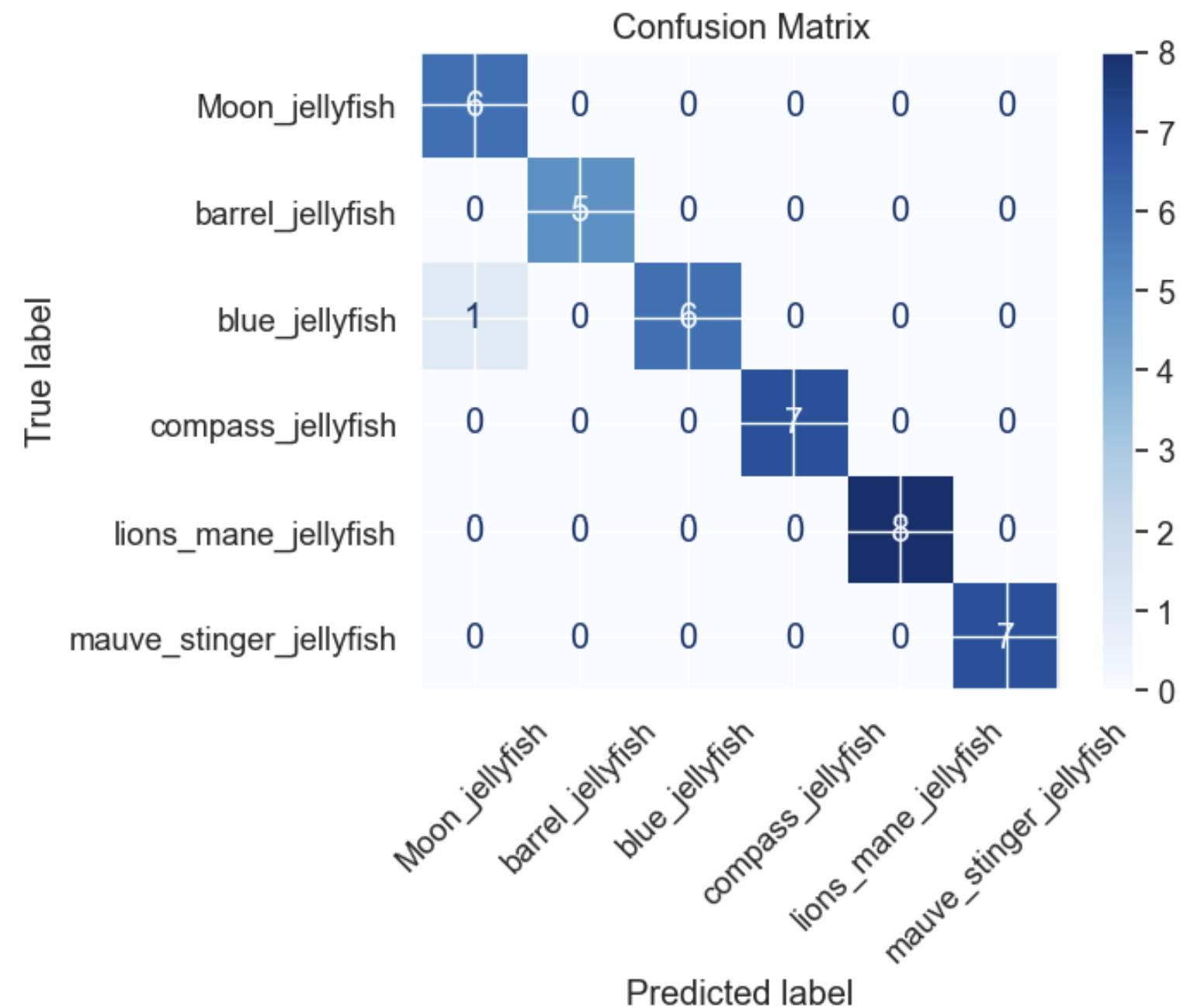
베이스 모델과 비교 포인트

항목	설명
장점	<ul style="list-style-type: none">- 복잡한 텍스처, 유사한 해파리 모양 등 고차원 시각 패턴 인식에 탁월- 데이터 양이 적더라도 전이학습 + fine-tuning으로 높은 성능 가능
베이스라인 대비 기대 효과	<ul style="list-style-type: none">- 유사 색상 / 형태 클래스에서도 미세한 패턴 학습 가능성 ↑- 정교한 피쳐 추출 가능

5.4 모델 개선 – ConvNext

5. 모델 개선

모델 개선 – ConvNext



5.4 모델 개선 – ConvNext

5.

모델 개선

모델 개선 – ConvNext

[결과 확인]

- 테스트 정확도: 0.9750, 테스트 손실: 0.2209 로, blue <-> moon 1개를 제외하고는 전부 정답.
 - **Train Accuracy**
 - 거의 2 epoch 이후부터 1.0에 수렴 -> 학습 데이터 과적합 가능성
 - Loss도 빠르게 0에 수렴, 오차 거의 없음
 - **Validation Accuracy**
 - 초기 상승 후 급락 。 이후 0.79~0.85 수준에서 수렴
 - Validation Loss도 거의 평평하게 유지됨
 - 👉 과적합 예상됨.
- 👉 유일한 오답을 내었던 blue-moon 의 데이터셋을 증강한다면 정확도 100%의 가능성 있음

5. 모델 개선

모델 개선 – ResNet50

베이스 모델과 비교 포인트

항목

설명

장점

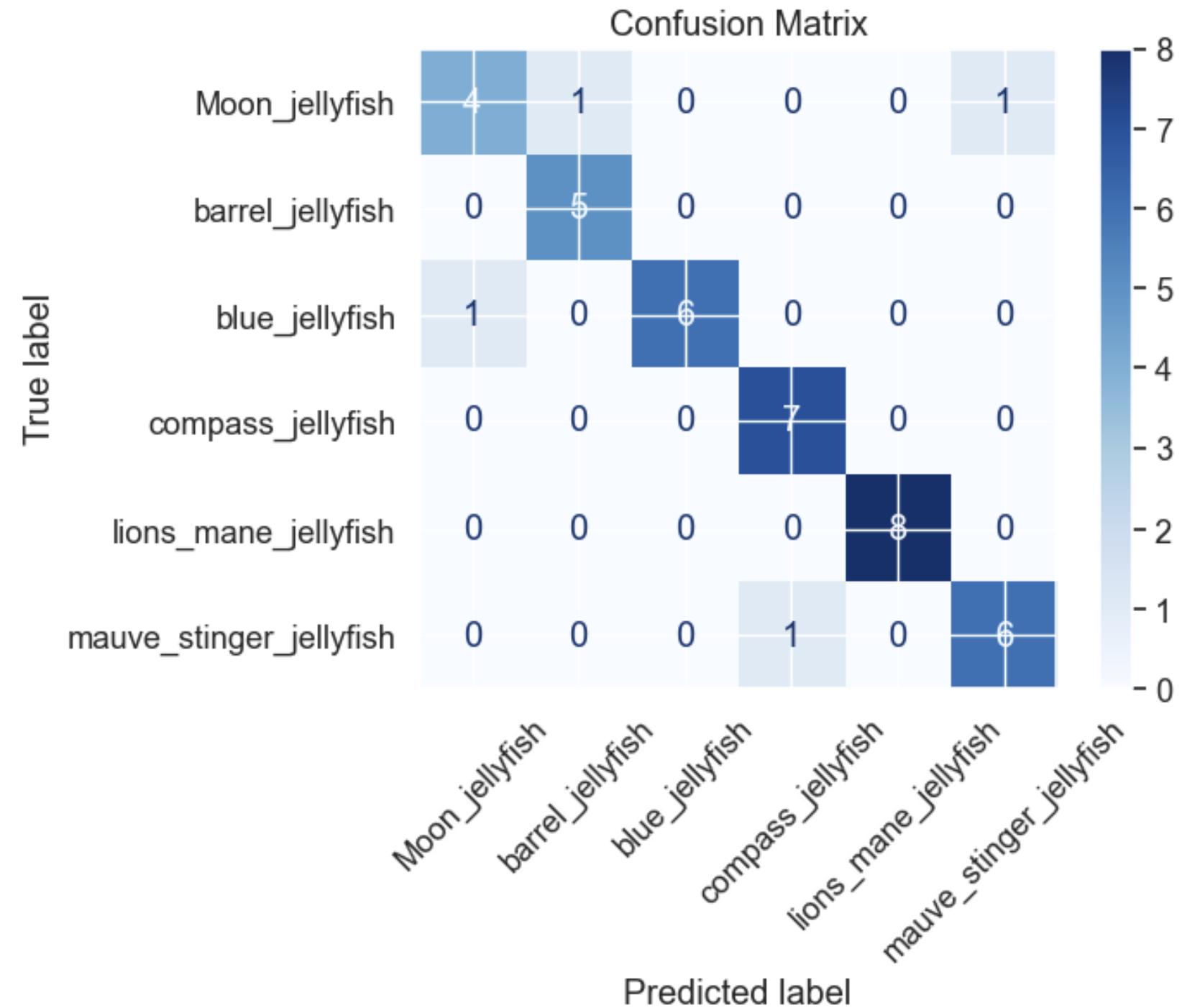
깊은 네트워크로 복잡한 패턴 학습 가능

베이스라인 대비 기대 효과

- 형태가 비슷하지만 차이가 미묘한 경우 → 깊은 네트워크가 차이 포착 가능성 높음

5. 모델 개선

모델 개선 – ResNet50



5.5 모델 개선 – ResNet50

5. 모델 개선

모델 개선 – ResNet50

[결과 확인]

- 테스트 정확도: 0.9000, 테스트 손실: 0.4557 로, 준수한 성능 확인
 - **Train Accuracy**
 - 거의 완벽하게 1.0에 수렴 -> 과적합 가능성 있음
 - **Validation Accuracy**
 - 초반 빠르게 상승 후 불안정한 패턴
 - epoch 증가에도 성능이 일정 이상 향상되지 않음 -> 학습이 일반화되지 않았음
 - 3~5 에포크부터 과적합 의심
- 👉 유일한 오답을 내었던 blue-moon 의 데이터셋을 증강한다면 정확도 100%의 가능성 있음

5. 모델 개선

모델 개선 – 결론

- 가장 준수한 성능을 보였던 ConvNext 및 Resnet50 모두 데이터셋 증강을 통해 정확도 향상의 여지가 있음을 확인함

[ConvNeXt가 1위를 차지한 이유]

1. 현대적 구조로 개선된 CNN

- ConvNeXt는 ResNet을 기반으로 하지만, Transformer의 설계를 차용해 성능을 높인 최신 CNN
- 커널 사이즈 확장(7x7), 더 깊은 레이어, LayerNorm 등의 도입으로 전통적인 CNN의 한계를 극복함

2. 더 깊고 넓은 구조

- 더 많은 파라미터와 깊이로 인해 복잡한 패턴도 더 잘 학습
- 이는 해파리 이미지처럼 종 간 경계가 모호한 데이터에서 높은 분류력을 발휘하는 데 유리

3. 효율적인 downsampling 및 normalization

- 전처리된 이미지의 특성을 잘 반영할 수 있는 구조임.
- 데이터 전처리(예: 정규화, 리사이징)가 잘 되어 있을수록 ConvNeXt의 성능이 더 극대화

6.

결론 및 한계점

결론 및 한계점

본 프로젝트는 해파리 이미지 데이터를 기반으로 종 분류 모델을 개발하는 것을 목표로 하였습니다. 다양한 CNN 기반 모델을 실험한 결과, ConvNeXt와 ResNet50이 각각 1위, 2위의 높은 정확도를 기록하였습니다.

ConvNeXt는 최신 구조의 CNN으로, 넓은 커널 사이즈, 향상된 normalization 기법, 깊은 네트워크 설계를 통해 복잡한 시각적 패턴을 효과적으로 학습하였습니다.

전처리와 EDA를 통해 데이터 품질을 높인 점이 모델 성능 향상에 핵심적인 역할을 했습니다.

ResNet50은 잔차 연결 구조를 활용한 안정적인 학습 능력과 범용성이 강점이었으며, 데이터에 대한 일반화 성능이 뛰어났습니다.

프로젝트 수행 과정에서 **EDA 및 전처리의 중요성**이 명확히 드러났으며, 데이터 불균형, 클래스 간 유사도, 이미지 품질 편차 등이 분류 성능에 직접적인 영향을 미치는 주요 요인임을 확인했습니다.

[한계 및 향후 방향]

- 일부 클래스에 대한 데이터 수 부족 및 이미지 품질 편차는 여전히 성능 저하 요인으로 작용함
- 향후 이미지 증강 적용을 통해 분류 정확도를 더욱 향상시킬 수 있음



감사합니다.

REPORT