PROJECT

CNN Dog or Cat Classification

dmc 이종진

2025.05.28

✳ Contents

✳ 프로젝트 개요

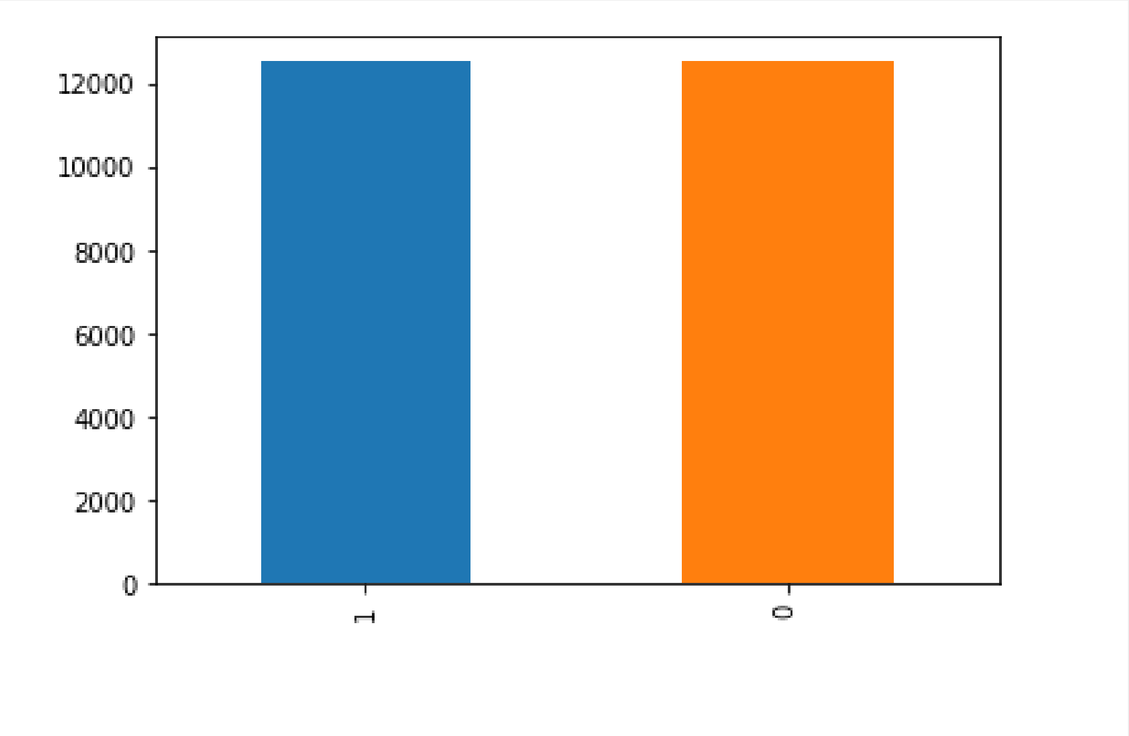
목적

이 프로젝트는 이미지 속 동물이 강아지인지 고양이인지 구분하는 분류 모델을 만들어보는 걸 목표로 합니다.
딥러닝에서 자주 쓰이는 CNN(Convolutional Neural Network) 구조를 직접 구현하면서, 이미지 분류가 실제로 어떻게 이뤄지는지 체험해보고자 했습니다.
데이터는 Kaggle에서 제공하는 개·고양이 이미지 데이터셋을 활용했고, 모델 구현은 TensorFlow/Keras 기반으로 진행했습니다.
처음에는 단순히 모델을 만들고 학습시키는 데 집중했지만, 점점 구조를 다듬고 하이퍼파라미터를 조정하면서 모델의 성능이 어떻게 달라지는지 보는 재미도 컸습니다.
실제 프로젝트처럼 훈련, 평가, 테스트, 제출까지의 전 과정을 직접 다뤄보면서 이진 분류(Binary Classification) 문제를 딥러닝 방식으로 푸는 경험을 해보는 게 주요 목표였습니다.

✳ 데이터셋 소개

Kaggle Dogs vs. Cats 이미지 데이터셋

- train.zip: 학습용 이미지가 포함되어 있으며, 각 이미지 파일 이름에는 카테고리 정보가 포함되어 있습니다.
- test1.zip: 테스트용 이미지가 들어 있으며, 카테고리는 제공되지 않습니다.
- sampleSubmission.csv: 테스트 결과를 어떤 형식으로 제출해야 하는지를 보여주는 예시 파일입니다.



✳ 데이터 전처리

카테고리 추출

파일 이름(예.dog.123.jpg/cat.456.jpg)에서 문자열을 기준으로 dog/cat 카테고리를 구분

데이터프레임 구성

이미지 파일명과 카테고리를 하나의 테이블로 정리

이미지 크기 정규화

다양한 해상도의 이미지를 모두 128x128 크기로 리사이즈

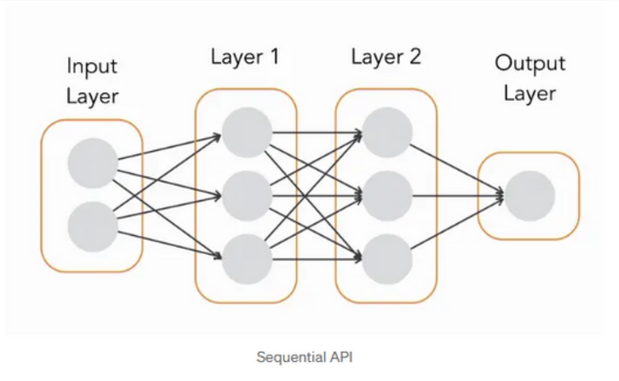
데이터 증강 적용

회전, 이동, 확대, 좌우 반전 등 적용하여 학습 데이터 다양성 확보

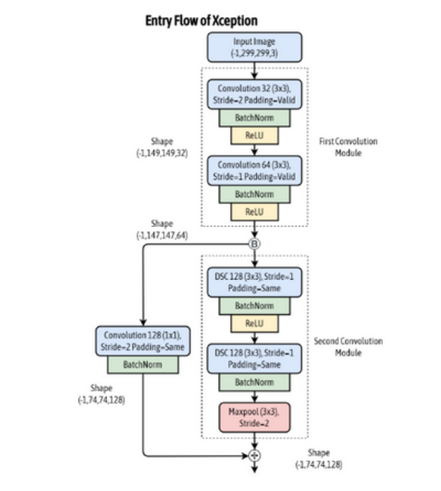
ImageDataGenerator 사용

학습 데이터는 증강 포함, 검증 데이터는 정규화만 적용하여 구성

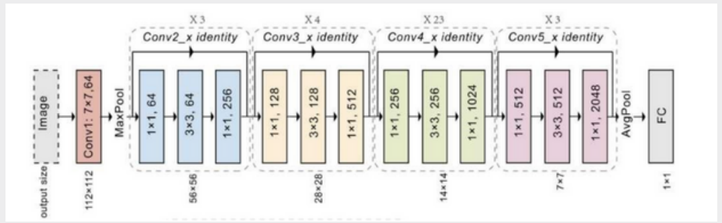
✳ 사용 모델



Sequential



Xception



ResNet50

✳ 모델 설계 Sequential

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 126, 126, 32)	896
batch_normalization (BatchNormalization)	(None, 126, 126, 32)	128
max_pooling2d (MaxPooling2D)	(None, 63, 63, 32)	0
dropout (Dropout)	(None, 63, 63, 32)	0
conv2d_1 (Conv2D)	(None, 61, 61, 64)	18,496
batch_normalization_1 (BatchNormalization)	(None, 61, 61, 64)	256
max_pooling2d_1 (MaxPooling2D)	(None, 30, 30, 64)	0
dropout_1 (Dropout)	(None, 30, 30, 64)	0
conv2d_2 (Conv2D)	(None, 28, 28, 128)	73,856
batch_normalization_2 (BatchNormalization)	(None, 28, 28, 128)	512
max_pooling2d_2 (MaxPooling2D)	(None, 14, 14, 128)	0
dropout_2 (Dropout)	(None, 14, 14, 128)	0
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 512)	12,845,568
batch_normalization_3 (BatchNormalization)	(None, 512)	2,048
dropout_3 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 2)	1,026

- 입력층 : 128x128 RGB 이미지 입력
- Conv Block1 :

  Conv2D(32) → BatchNorm → MaxPooling2D → Dropout(0.25)
- Conv Block2:

  Conv2D(64) → BatchNorm → MaxPooling2D → Dropout(0.25)
- Conv Block3:

  Conv2D(128) → BatchNorm → MaxPooling2D → Dropout(0.25)
- 완전 연결층 :

  Flatten → Dense(512, ReLU) → BatchNorm → Dropout(0.5)
- 출력층 : Dense(2, Softmax)
- 손실 함수 : categorical_crossentropy
- 최적화 기법 : RMSprop
- 평가 지표 : Accuracy
- 콜백 설정 :

  EarlyStopping(patience=10), ReduceLROnPlateau(val_acc 기준,factor=0.5)

✳ 모델 설계 Xception 기반 전이 학습 모델

```
base_model = Xception(weights='imagenet', include_top=False, input_shape=(IMG_WIDTH, IMG_HEIGHT, 3))

base_model.summary()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/xception/xception_weights_tf_dim_ordering_tf_kernels_notop.h5
83683744/83683744 ━━━━━━━━━━━━━━━━━━━━ 1s 0us/step
Model: "xception"
```

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 224, 224, 3)	0	-
block1_conv1 (Conv2D)	(None, 111, 111, 32)	864	input_layer[0][0]
block1_conv1_bn (BatchNormalizatio…	(None, 111, 111, 32)	128	block1_conv1[0][…
block1_conv1_act (Activation)	(None, 111, 111, 32)	0	block1_conv1_bn[…
block1_conv2 (Conv2D)	(None, 109, 109, 64)	18,432	block1_conv1_act…
block1_conv2_bn (BatchNormalizatio…	(None, 109, 109, 64)	256	block1_conv2[0][…
block1_conv2_act (Activation)	(None, 109, 109, 64)	0	block1_conv2_bn[…
block2_sepconv1 (SeparableConv2D)	(None, 109, 109, 128)	8,768	block1_conv2_act…
block2_sepconv1_bn (BatchNormalizatio…	(None, 109, 109, 128)	512	block2_sepconv1[…
block2_sepconv2_act (Activation)	(None, 109, 109, 128)	0	block2_sepconv1_…
block2_sepconv2	(None, 109, 109,	17,536	block2_sepconv2_…

- 입력층 : 224x224 RGB 이미지 입력
- Base Model :

  Xception (ImageNet 사전 학습 가중치 사용, include_top=False)
- Feature Extractor :

  GlobalAveragePooling2D()
- 완전 연결층 :

  Dense(16, ReLU) → Dropout(0.25)
- 출력층 : Dense(2, Softmax)
- 손실 함수 : categorical_crossentropy
- 최적화 기법 : Adam(learning_rate=0.00002)
- 평가 지표 : Accuracy
- 콜백 설정 :

  EarlyStopping(monitor='val_loss', patience=5)

  ModelCheckpoint(monitor='val_loss', save_best_only=True)

✳ 모델 설계 ResNet50 기반 전이 학습 모델

```
rbase_model = ResNet50(weights='imagenet', include_top=False, input_shape=(IMG_WIDTH, IMG_HEIGHT, 3))
rbase_model.trainable = False

res_model = models.Sequential([
    rbase_model,
    layers.GlobalMaxPooling2D(),
    layers.Dense(256, activation='relu'),
    layers.Dropout(0.25),
    layers.Dense(2, activation='softmax')
])
res_model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

res_model.summary()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/resnet/resnet50_weight
94765736/94765736 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
Model: "sequential_1"
```

Layer (type)	Output Shape	Param #
resnet50 (Functional)	(None, 7, 7, 2048)	23,587,712
global_max_pooling2d (GlobalMaxPooling2D)	(None, 2048)	0
dense_2 (Dense)	(None, 256)	524,544
dropout_1 (Dropout)	(None, 256)	0
dense_3 (Dense)	(None, 2)	514

```
Total params: 24,112,770 (91.98 MB)
Trainable params: 525,058 (2.00 MB)
Non-trainable params: 23,587,712 (89.98 MB)
```

- 입력층 : 224x224 RGB 이미지 입력
- Base Model :

  ResNet50 (ImageNet 사전 학습 가중치 사용, include_top=False)

  - base_model.trainable = False로 설정하여 가중치 고정
- Feature Extractor :

  GlobalMaxPooling2D()
- 완전 연결층 :

  Dense(256, ReLU) → Dropout(0.25)
- 출력층 : Dense(2, Softmax)
- 손실 함수 : categorical_crossentropy
- 최적화 기법 : Adam
- 평가 지표 : Accuracy
- 콜백 설정 :

  EarlyStopping(monitor='val_loss', patience=5)

  ModelCheckpoint(monitor='val_loss', save_best_only=True)

✳ 모델 컴파일 및 하이퍼파라미터 설정

기본 CNN 모델

손실 함수	categorical_crossentropy
최적화 기법	RMSprop
평가 지표	Accuracy
학습률	(기본값)
배치 사이즈	32
에폭 수	50
콜백 설정	EarlyStopping(patience=10), ReduceLROnPlateau(monitor='val_acc', factor=0.5)

✳ 모델 컴파일 및 하이퍼파라미터 설정

Xception 기반 전이 학습 모델

손실 함수	categorical_crossentropy
최적화 기법	Adam(learning_rate=0.00002)
평가 지표	Accuracy
배치 사이즈	32
에폭 수	30
콜백 설정	EarlyStopping(monitor='val_loss', patience=5) ModelCheckpoint(save_best_only=True)

✳ 모델 컴파일 및 하이퍼파라미터 설정

ResNet50 기반 전이 학습 모델

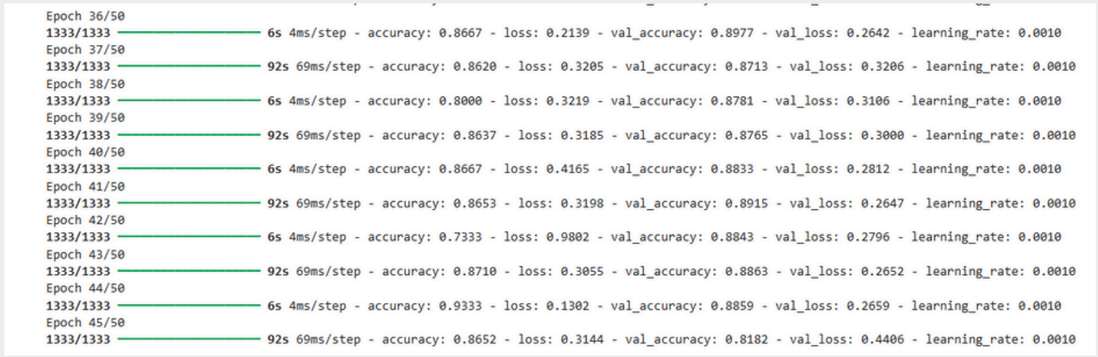
손실 함수	categorical_crossentropy
최적화 기법	Adam(기본 학습률 사용)
평가 지표	Accuracy
배치 사이즈	32
에폭 수	30
콜백 설정	EarlyStopping(monitor='val_loss', patience=5) ModelCheckpoint(save_best_only=True)

✳ 모델 학습

기본 CNN 모델

전체 데이터셋을 기준으로 학습용 데이터와 검증용 데이터를 미리 분리한 후, 각각 ImageDataGenerator를 통해 불러와 사용하였습니다.
학습 데이터는 회전, 확대, 이동, 뒤집기 등의 다양한 이미지 증강 기법을 적용하였고, 검증 데이터는 정규화만 수행하였습니다.

train_generator에는 학습용 20,000장의 이미지가,
validation_generator에는 검증용 5,000장의 이미지가 포함되어 있습니다.
이렇게 구성된 두 generator를 활용하여 모델의 학습과 성능 평가를 진행하였습니다.



✳ 모델 학습

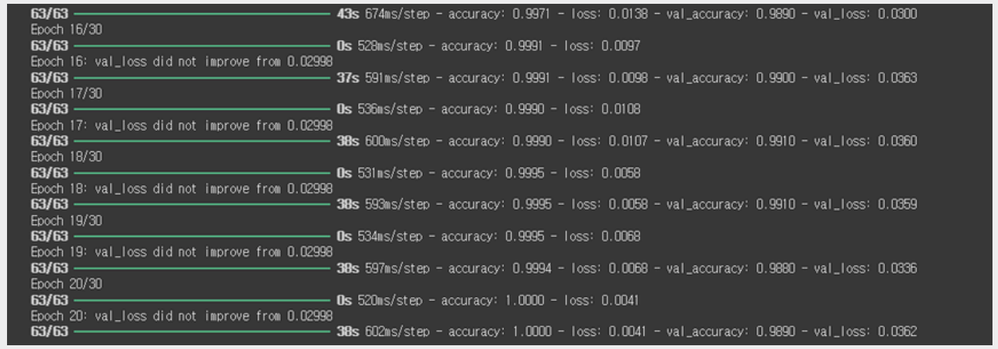
Xception & ResNet50

Xception 및 ResNet50 기반 모델은 디렉토리 구조로 분리된 학습용(train)과 검증용(validation) 데이터를 `flow_from_directory()` 함수를 통해 불러와 학습에 사용하였습니다.

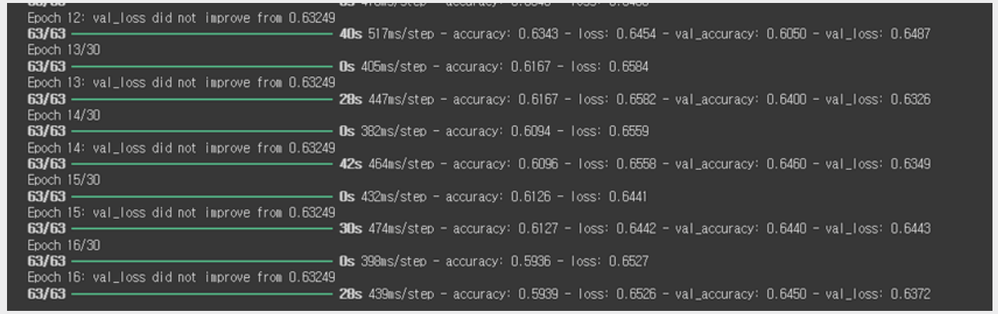
학습 데이터에는 회전, 이동, 확대 등의 이미지 증강을 적용하였고, 검증 데이터에는 정규화만 수행하였습니다.
모든 이미지는 224×224 크기로 리사이즈되었으며, RGB 채널 형식으로 로딩하였습니다.

제너레이터는 각각 32의 배치 크기로 구성하였으며, 에폭마다 학습 데이터는 무작위로 셔플되어 모델의 일반화 성능 향상에 기여하였습니다.

Xception



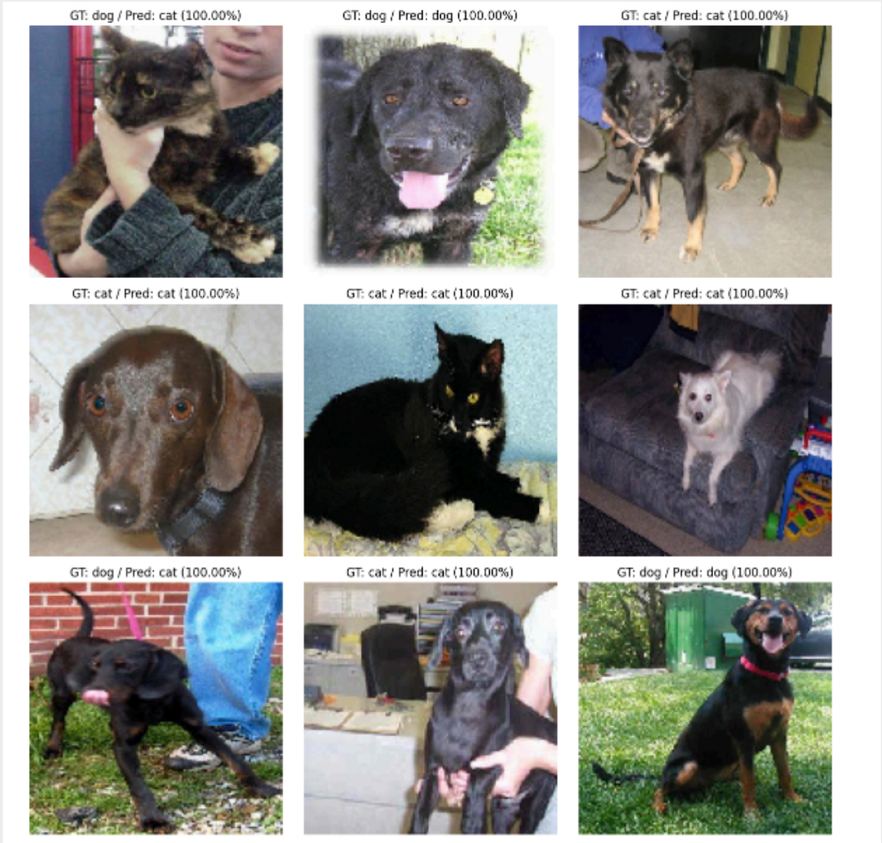
ResNet50



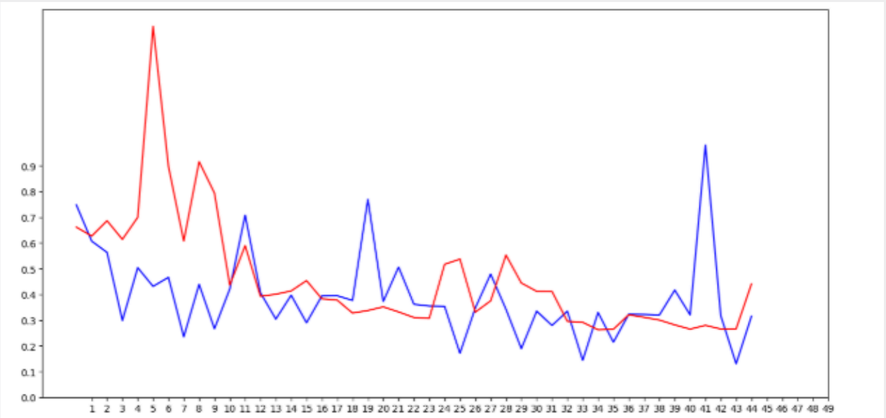
✳ 모델 성능 평가 및 테스트 예측

기본 CNN 모델

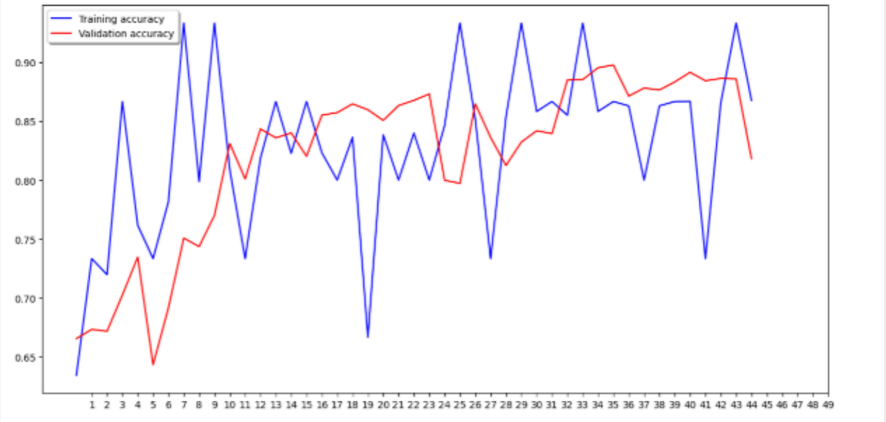
- Acuracy : 0.8182
- Loss : 0.4406



Loss



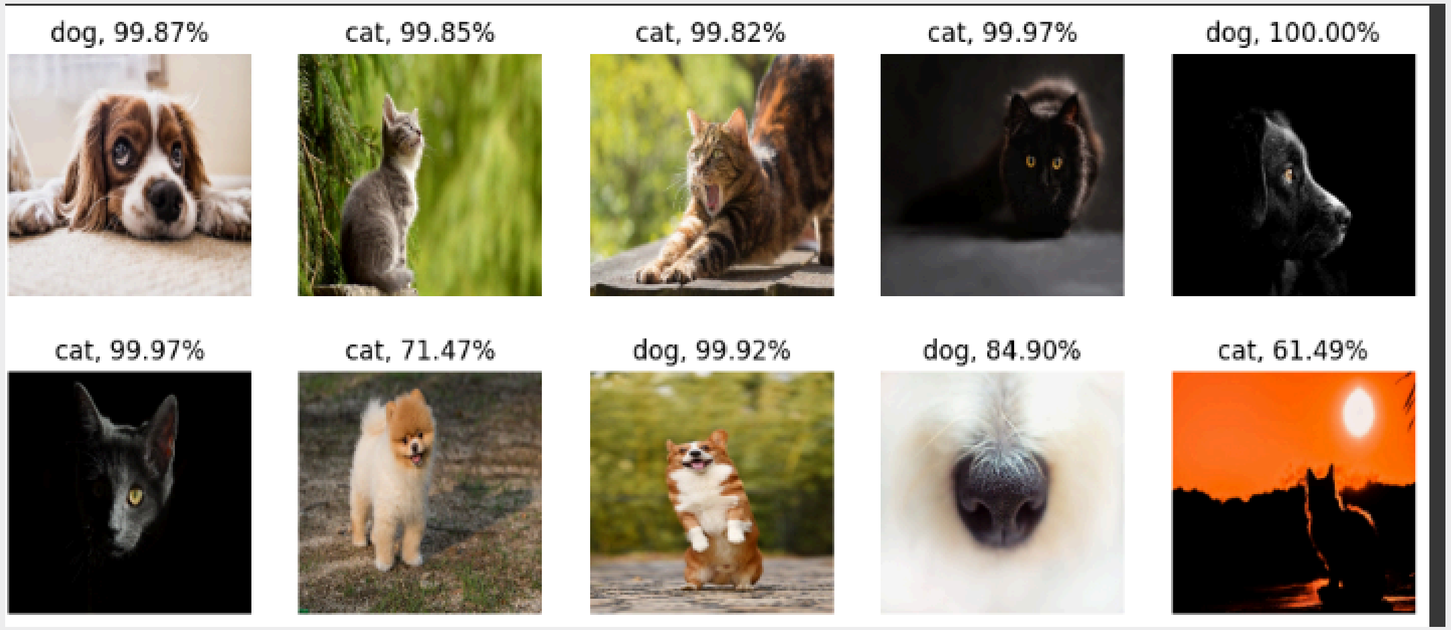
Accuracy

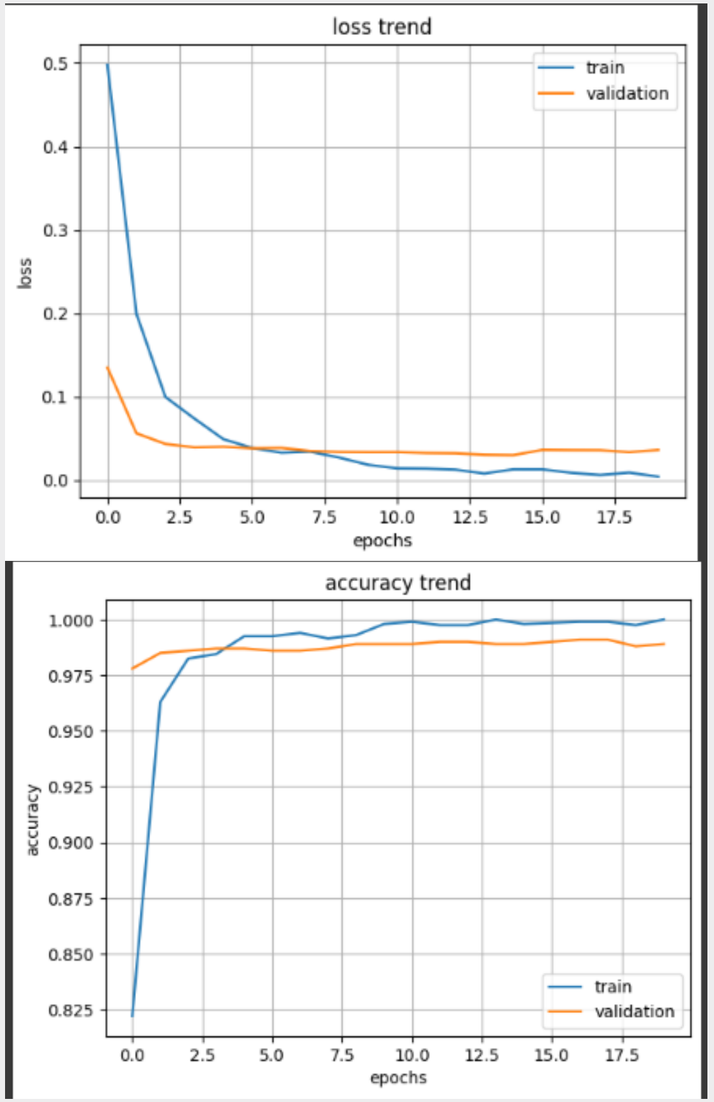


✳ 모델 성능 평가 및 테스트 예측

Xception

- Acuracy : 0.9909
- Loss : 0.0299

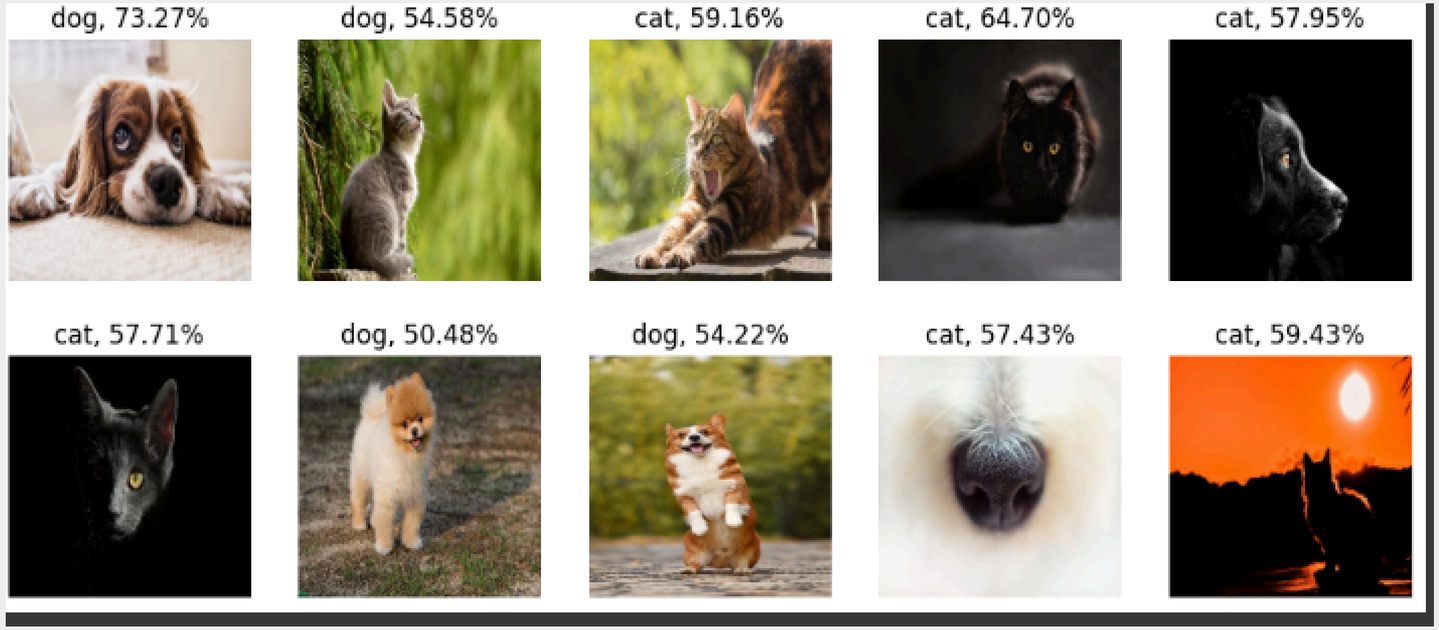


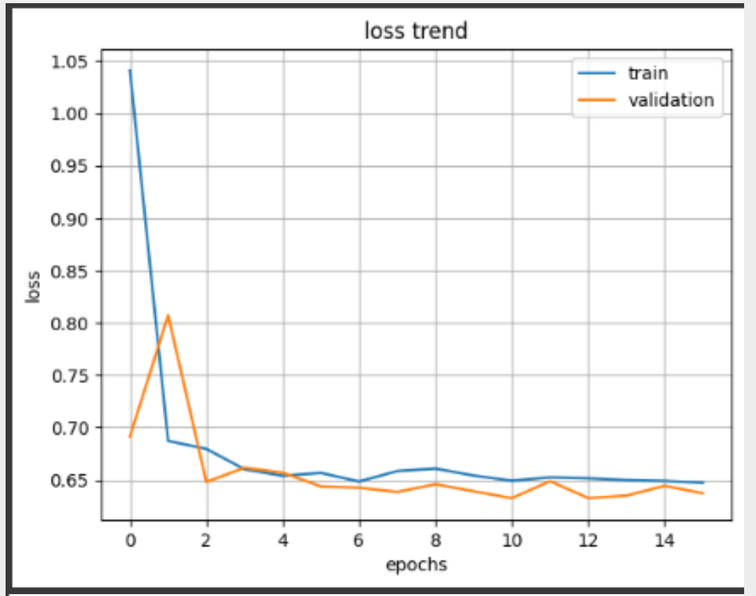


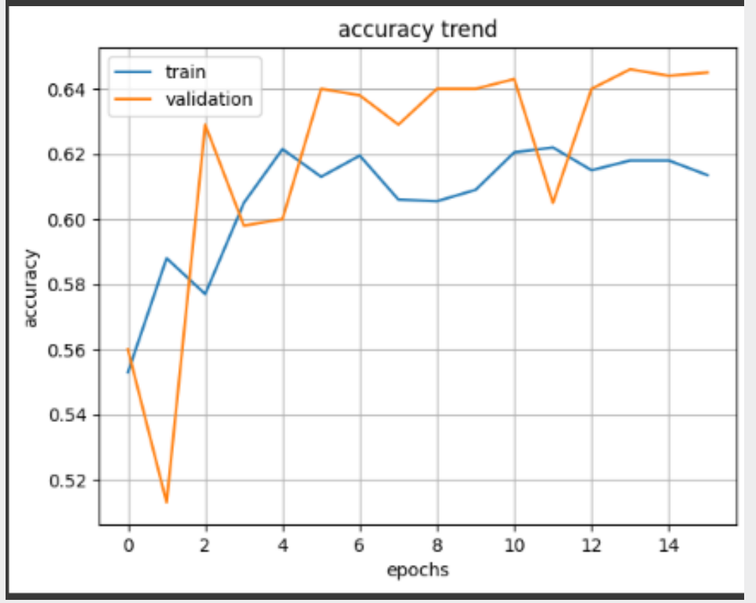
✳ 모델 성능 평가 및 테스트 예측

ResNet50

- Acuracy : 0.5010
- Loss : 0.8620







✳ 결론

기본 CNN 모델

비교적 단순한 구조로도 일정 수준 이상의 정확도를 달성했지만, overfitting 현상과 예측 일관성 부족이 관찰됨.

Xception 기반 모델

사전 학습된 고성능 모델을 활용해 빠른 수렴과 높은 정확도를 달성함. 학습 곡선도 매우 안정적이며 과적합 없이 학습됨.

ResNet50 기반 모델

사전 학습된 ResNet50 모델을 기반으로 학습을 진행했으나, 과적합 또는 학습률 설정 문제 등으로 인해 성능이 낮게 나타났습니다.
훈련 정확도는 상승하는 반면 검증 정확도는 불안정한 모습을 보였고, 일부 샘플에서 예측 확신도도 낮게 나타남

✳ 개선 방향

1. 과적합 방지 강화
   - Dropout 비율 조정, L2 정규화, 데이터 증강 확대 등을 통해 일반화 성능 향상

2. 테스트 데이터에 대한 정량적 평가
   - Precision, Recall, F1-score 등의 평가 지표 추가 적용

3. 경량 모델 실험
   - MobileNet, EfficientNet 등 경량 아키텍처를 통한 추론 속도 및 실용성 확보

4. 예측 결과 시각화
   - Grad-CAM 등 시각화 기법 도입을 통해 모델의 판단 근거를 해석 가능하게 함

5. 데이터셋 확장 및 다중 클래스 분류
   - 다양한 동물 종 분류 문제로 확장하여 모델의 범용성과 확장성 검토

감사합니다.