AI 라이브러리 활용

프로젝트 발표

개와 고양이 이미지 사진 분류

20210710 맹지우

2024. 11. 25.

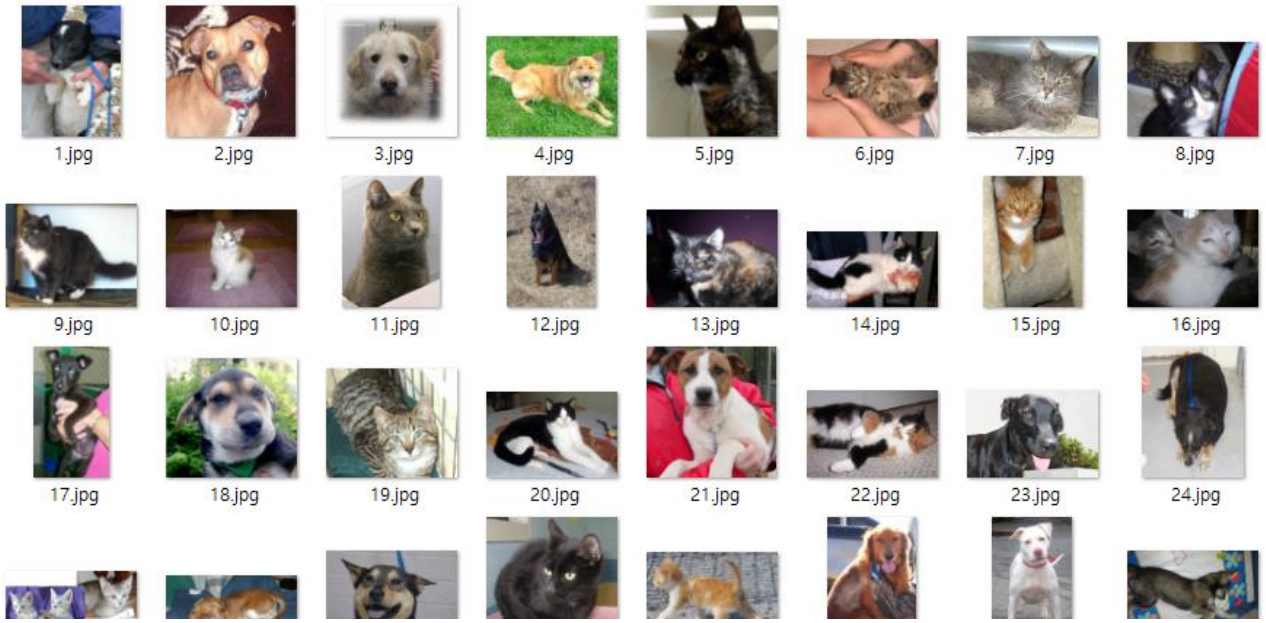
프로젝트 배경

- 개와 고양이 이미지 분류 문제는 딥러닝 입문자들에게 적합한 주제로, CNN의 개념을 이해하고 실습을 통해 학습 과정을 좋은 문제라고 생각함.
- Kaggle 데이터셋을 기반으로 이미지 분류 모델을 학습하고, 데이터 증강 기법을 적용해 성능을 향상시키는 것이 목표.

1. 학습을 위한 데이터 준비

이미지 처리를 위해 학습 데이터가 필요

기존에 있는 학습 데이터를 이용하여 학습 및 평가 진행



출처: https://www.kaggle.com/competitions/dogs-vs-cats/data

2. 학습을 위해 모델 생성

```
model = Sequential()

model.add(Conv2D(32, (3,3), activation='relu', input_shape=(128, 128, 3)))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Conv2D(64, (3,3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Conv2D(128, (3,3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))
model.add(Dense(2, activation='softmax'))

model.compile(loss='categorical_crossentropy', optimizer='rmsprop', metrics=['accuracy'])
```

2. 학습을 위해 모델 생성

- input_shape는 (Width, Height, channels) 값
- 컬러 이미지이기 때문에 RGB 3채널인 모습 확인

```
model = Sequential()

model.add(Conv2D(32, (3,3), activation='relu', input_shape=(128, 128, 3)))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Conv2D(64, (3,3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Conv2D(128, (3,3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))
model.add(Dense(2, activation='softmax'))

model.compile(loss='categorical_crossentropy', optimizer='rmsprop', metrics=['accuracy'])
```

- Convolution을 통해 filter로 이미지 특징 추출
- Pooling을 통해 이미지 크기는 축소
  (단, 이미지 특징은 유지)
- 각 층의 활성함수는 RELU를 사용
- 출력층의 활성함수는 softmax 사용

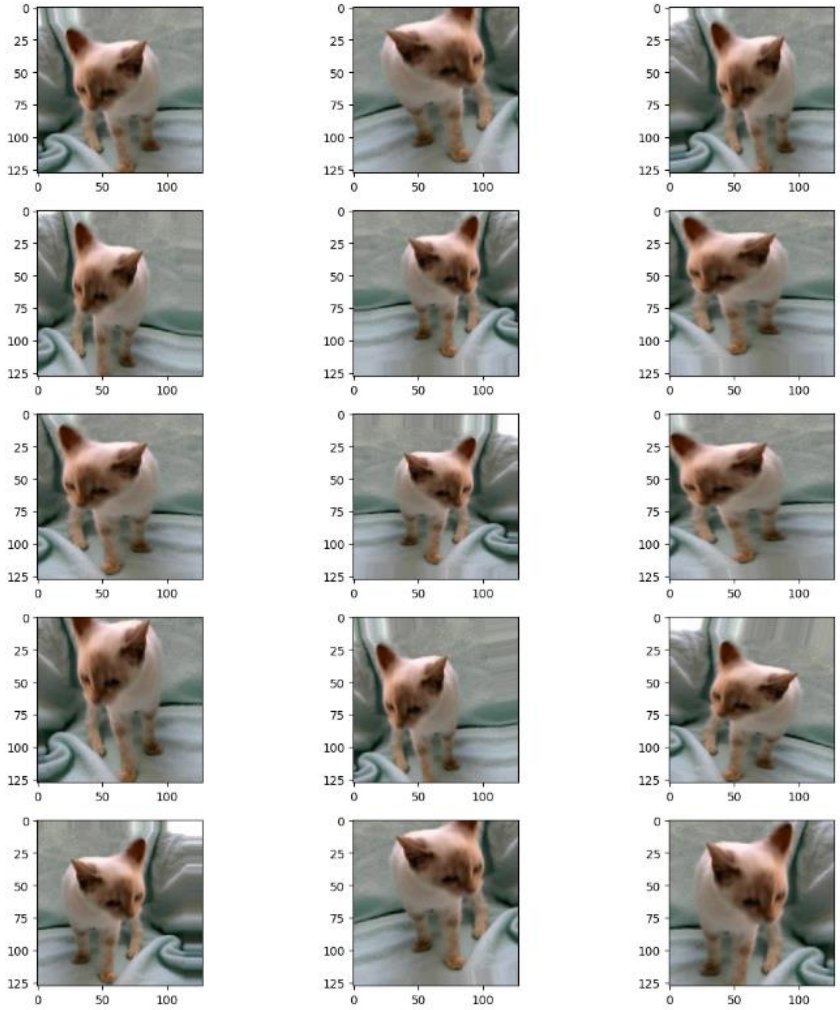
3. 데이터 증강 및 학습 데이터 생성

```
train_datagen = ImageDataGenerator(
    rotation_range=15,
    rescale=1./255,
    shear_range=0.1,
    zoom_range=0.2,
    horizontal_flip=True,
    width_shift_range=0.1,
    height_shift_range=0.1
)

train_generator = train_datagen.flow_from_dataframe(
    train_df,
    "C:/Users/ADM/Desktop/b/train/",
    x_col='filename',
    y_col='category',
    target_size=IMAGE_SIZE,
    class_mode='categorical',
    batch_size=batch_size
)
```

- 학습 데이터의 수가 많으면 많을수록 학습이 잘 이루어질 수 있음
- 따라서 한 이미지를 zoom하거나 상하 또는 좌우 반전 등을 통해 변화를 주어 이미지 증식

3. 데이터 증강 및 학습 데이터 생성

```
train_datagen = ImageDataGenerator(
    rotation_range=15,
    rescale=1./255,
    shear_range=0.1,
    zoom_range=0.2,
    horizontal_flip=True,
    width_shift_range=0.1,
    height_shift_range=0.1
)

train_generator = train_datagen.flow_from_dataframe(
    train_df,
    "C:/Users/ADM/Desktop/b/train/",
    x_col='filename',
    y_col='category',
    target_size=IMAGE_SIZE,
    class_mode='categorical',
    batch_size=batch_size
)
```



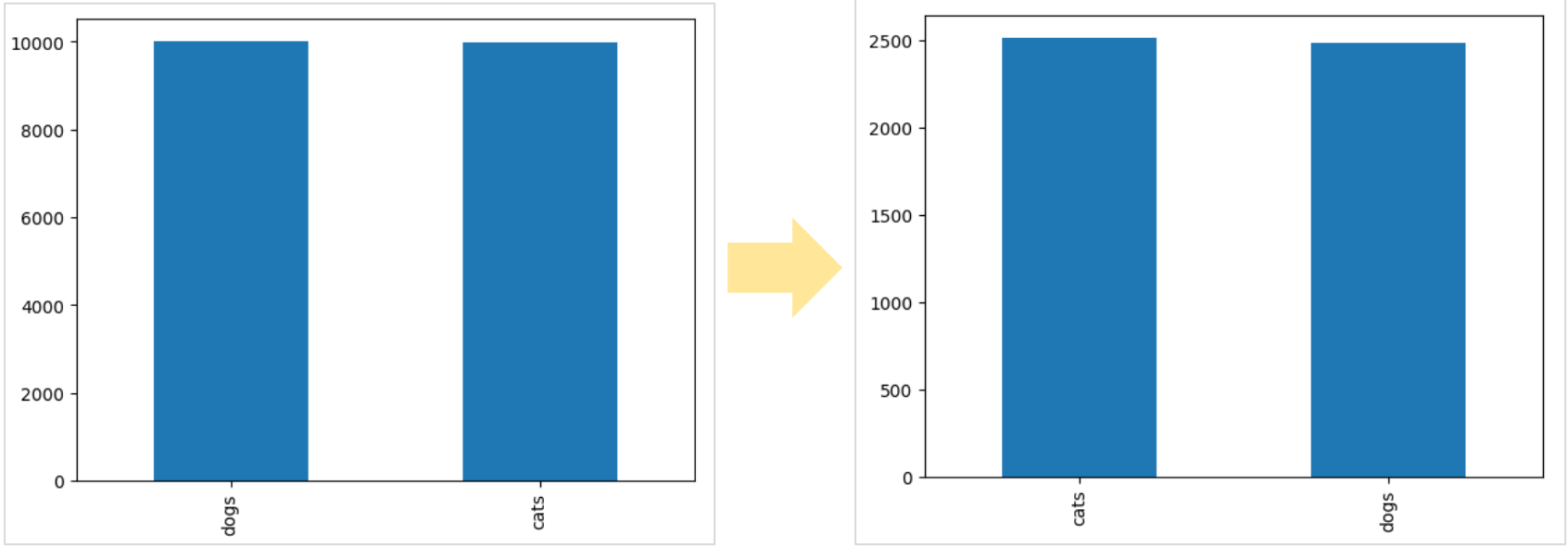
4. 학습 및 검증 데이터셋 구성 과정

```
df["category"] = df["category"].replace({0: 'cats', 1: 'dogs'})


train_df, validate_df = train_test_split(df, test_size=0.20, random_state=42)
train_df = train_df.reset_index(drop=True)
validate_df = validate_df.reset_index(drop=True)
```

- 일반적으로 검증을 위한 데이터는 학습 데이터의 20%를 사용
- 따라서 이에 맞춰 10000개의 데이터의 20%인 2500개의 데이터를 검증 데이터로 사용

4. 학습 및 검증 데이터셋 구성 과정

```
df["category"] = df["category"].replace({0: 'cats', 1: 'dogs'})


train_df, validate_df = train_test_split(df, test_size=0.20, random_state=42)
train_df = train_df.reset_index(drop=True)
validate_df = validate_df.reset_index(drop=True)
```

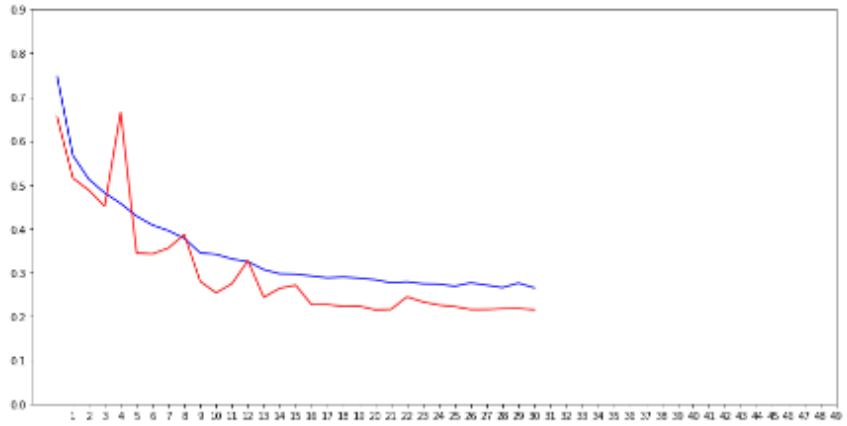


5. epochs 지정 후 학습을 진행

```
epochs=50 if FAST_RUN else 50
history = model.fit_generator(
    train_generator,
    epochs=epochs,
    validation_data=validation_generator,
    validation_steps=total_validate//batch_size,
    steps_per_epoch=total_train//batch_size,
    callbacks=callbacks
)
```

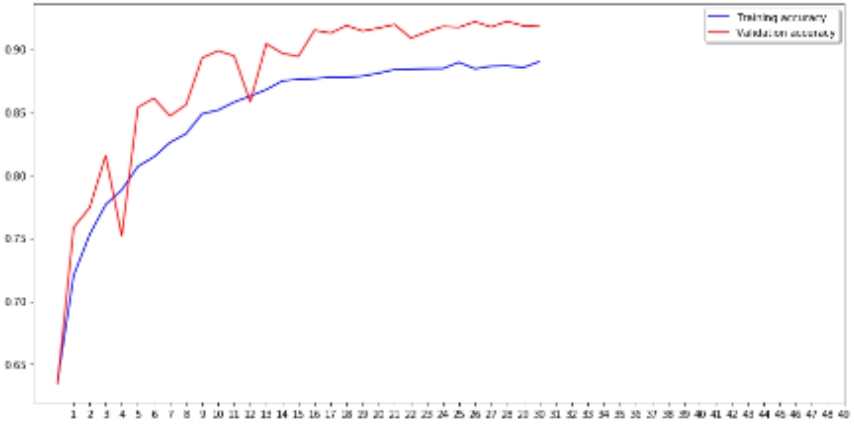
결론

빨간색 - 학습
파란색 - 검증



오답률

초반엔 오답률이 높지만 학습을 진행함에 따라 낮아지는 모습 확인



정확도

오답률과 반대로 초반엔 정확도가 낮지만 학습을 진행함에 따라 높아지는 모습 확인

THE END

감 사 합 니 다