9주차

https://www.kaggle.com/competitions/cifar-10/overview

- 주제 : 이미지를 10개의 클래스로 분류하기
- 데이터 : 10개의 클래스에 걸쳐 총 60,000개의 32x32 컬러 이미지로 구성된 이미지 분류 데이터셋

1. 데이터 전처리

- 정규화 : CIFAR-10 데이터의 픽셀 값을 정규화하여 모델 학습을 안정화
- 레이블 변환 : CIFAR-10의 정수형 레이블을 One-hot 인코딩하여 다중 클래스 분류에 적합한 형식으로 변환

```
# 데이터 정규화
train_data = train_data / 255.0
test_data = test_data / 255.0

# 레이블을 One-hot 인코딩
from tensorflow.keras.utils import to_categorical
train_labels = to_categorical(train_labels, num_classes=10)
test_labels = to_categorical(test_labels, num_classes=10)
```

2. CNN 모델 설계

- CIFAR-10 데이터셋의 이미지(32x32×3)를 처리하기 위한 CNN(Convolutional Neural Network) 설계
- 구성 요소
  - Conv2D : 이미지 특징 추출(32개, 64개의 필터 사용)
  - MaxPooling2D : 특징 맵을 다운샘플링하여 중요한 정보 유지
  - Dropout : 과적합 방지를 위해 일부 뉴런 비활성화

- Dense : 최종 출력층에 10개의 뉴런을 사용하여 CIFAR-10의 10개 클래스를 분류

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Fla

model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32
    MaxPooling2D((2, 2)),
    Dropout(0.25),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Dropout(0.25),
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(10, activation='softmax')
])

# 모델 요약
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d (MaxPooling2D)	(None, 15, 15, 32)	0
dropout (Dropout)	(None, 15, 15, 32)	0
conv2d_1 (Conv2D)	(None, 13, 13, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(None, 6, 6, 64)	0
dropout_1 (Dropout)	(None, 6, 6, 64)	0
flatten (Flatten)	(None, 2304)	0
dense (Dense)	(None, 128)	295,040
dropout_2 (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 10)	1,290

Total params: 315,722 (1.20 MB)
Trainable params: 315,722 (1.20 MB)
Non-trainable params: 0 (0.00 B)

3. 모델 컴파일

- 손실 함수 : categorical_crossentropy를 사용하여 다중 클래스 분류에 최적화

- 최적화 기법 : Adam 옵티마이저를 사용해 학습 속도를 높이고 효율적인 수렴 보장
- 평가지표 : accuracy를 사용하여 학습 정확도를 모니터링

```
from tensorflow.keras.optimizers import Adam

# 모델 컴파일
model.compile(optimizer=Adam(learning_rate=0.001),
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

4. 모델 학습

- 학습 데이터를 사용해 모델을 학습하고, 검증 데이터를 사용해 모델의 성능을 평가
- Epoch : 학습 데이터셋을 10번 반복 학습
- Batch Size : 한 번에 64개의 샘플로 학습

```
# 모델 학습
history = model.fit(train_data.reshape(-1, 32, 32, 3), train_
                    epochs=10,
                    batch_size=64,
                    validation_data=(test_data.reshape(-1, 32
```

5. 학습 결과 요약

- 학습 과정
  - 모델이 학습 데이터를 사용하여 특징을 학습
  - 검증 데이터를 통해 학습 중 과적합 판단
- 출력
  - 학습 및 검증 정확도와 손실이 Epoch마다 출력
  - 학습 곡선을 통해 성능을 시각화, 분석

```
import matplotlib.pyplot as plt
```

```
plt.plot(history.history['accuracy'], label='Training Accurac
plt.plot(history.history['val_accuracy'], label='Validation A
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```