두산 Rokey Boot Camp

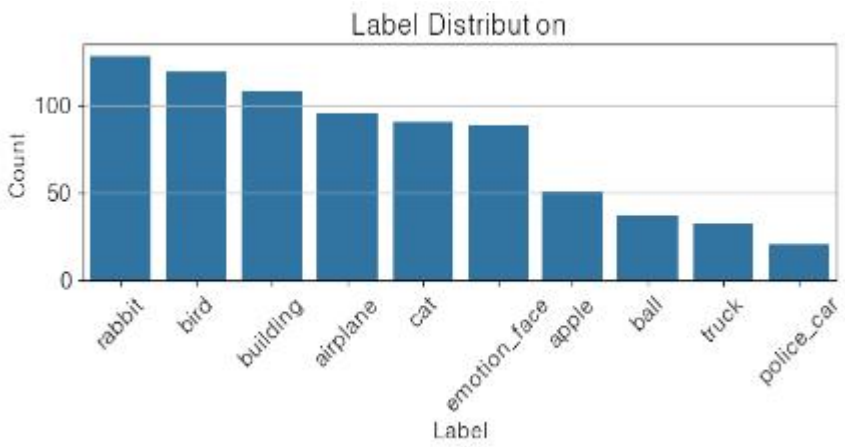
스터디 주간 활동 보고서

팀명	벌꿀오소리	**제출자 성명**	임소정
참여 명단	임소정, 송주훈, 강인우, 정민섭		
모임 일시	2025년 03월 28일 21시 ~22시 (총 1시간)		
장소	온라인(구글미트)	**출석 인원**	4/6
학습목표	데이콘에서 주관하는 이미지 분류 해커톤: 데이터 속 아이콘의 종류를 맞혀라! 대회 규칙에 부합하는 AI알고리즘을 개발할 수 있다.		
학습내용	송주훈 https://github.com/ChanwonJung/ROKEY_team6_study/blob/main/juhunsong/3.27%EC%8A%A4%ED%84%B0%EB%94%94.ipynb		

```python
# 2. CNN 모델 정의
class CNNFeatureExtractor(nn.Module):
    def __init__(self):
        super(CNNFeatureExtractor, self).__init__()
        self.conv_layers = nn.Sequential(
            nn.Conv2d(1, 32, kernel_size=3, stride=1, padding=1),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Conv2d(32, 64, kernel_size=3, stride=1, padding=1),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2)
        )
        self.fc = nn.Linear(64 * 8 * 8, 256)  # Feature Vector (256차원)

    def forward(self, x):
        x = self.conv_layers(x)
        x = x.view(x.size(0), -1)
        x = self.fc(x)
        return x
```

```python
# 3. CNN을 이용한 Feature 추출
cnn = CNNFeatureExtractor()
def extract_features(model, X_data):
    model.eval()
    X_tensor = torch.tensor(X_data, dtype=torch.float32)
    with torch.no_grad():
        features = model(X_tensor).numpy()
    return features

X_train_features = extract_features(cnn, X_train)
X_valid_features = extract_features(cnn, X_valid)
X_test_features = extract_features(cnn, X_test)
```

강인우https://ivy-cave-ab6.notion.site/03-27-1c322b700e8380298a20cf158f89ff04



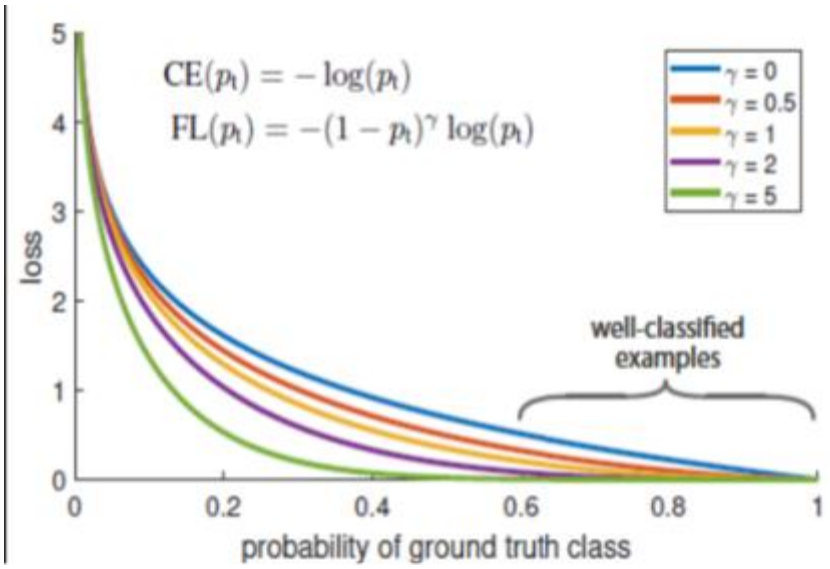
```
class model 3(nn.Module):
    def __init__(self, num_classes):
        super().__init__()
        self.relu = nn.ReLU(inplace=True)
        self.maxpool = nn.MaxPool2d(2, 2)
        self.flatten = nn.Flatten()

        self.features = nn.Sequential(
            nn.Conv2d(1, 32, 3, padding=1),
            nn.BatchNorm2d(32),
            self.relu,
            nn.Conv2d(32, 32, 3, padding=1),
            nn.BatchNorm2d(32),
            self.relu,
            self.maxpool,
            nn.Dropout(0.2),

            nn.Conv2d(32, 64, 3, padding=1),
            nn.BatchNorm2d(64),
            self.relu,
            nn.Conv2d(64, 64, 3, padding=1),
            nn.BatchNorm2d(64),
            self.relu,
            self.maxpool,
            nn.Dropout(0.3),

            nn.Conv2d(64, 128, 3, padding=1),
            nn.BatchNorm2d(128),
            self.relu,
            nn.Conv2d(128, 128, 3, padding=1),
            nn.BatchNorm2d(128),
            self.relu,
            self.maxpool,
            nn.Dropout(0.4)
        )

        self.classifier = nn.Sequential(
            nn.Linear(128 * 4 * 4, 256),
            self.relu,
            nn.Dropout(0.4),
            nn.Linear(256, num_classes)
        )
```

정민섭

https://github.com/ChanwonJung/ROKEY_team6_study/blob/main/MINSEOP_JEONG/Study_week_9_resnet18_with_FocalLoss.ipynb

FocalLoss

FocalLoss란? Object Detection에서 Background / Foreground Class의 불균형 문제를 loss 함수로 해결하기 위해 제안된 loss



FocalLoss
함수로 해결하기 위해 제안된
loss

```
import torch
import torch.nn as nn

class FocalLoss(nn.Module):
    def __init__(self, alpha=0.25, gamma=2, reduce=True):
        super(FocalLoss, self).__init__()
        self.alpha = alpha
        self.gamma = gamma
        self.reduce = reduce

    def forward(self, inputs, targets):
        ce_loss = nn.CrossEntropyLoss(reduction='none')(inputs, targets)  # Fix

        pt = torch.exp(-ce_loss)
        focal_loss = self.alpha * (1 - pt) ** self.gamma * ce_loss

        if self.reduce:
            return focal_loss.mean()
        else:
            return focal_loss
```

Model & Optim Setup

```
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

model = resnet18(pretrained=False)
model.fc = nn.Linear(model.fc.in_features, 10)
model = model.to(device)

# criterion = nn.CrossEntropyLoss()
criterion = FocalLoss(alpha=0.25, gamma=2.0)
optimizer = optim.AdamW(model.parameters(), lr=LR) # torch.optim.Adam(para
scheduler = optim.lr_scheduler.CosineAnnealingLR(optimizer, T_max=N_EPOCHS)
```

AdamW란? Adam 옵티마이저의 변형으로, 가중치 감쇠(weight decay)를 적용하는 것이
가중치 감쇠는 모델의 가중치를 감소시킴으로써 모델의 복잡성을 제어하여, 오버피팅(ov
완화하는 효과가 있습니다.

- params: 최적화할 모델의 파라미터들
- lr: 학습률(learning rate), 기본값은 0.001
- betas: 긴마 값들 (beta1, beta2)로 이루어진 튜플, 기본값은 (0.9, 0.999)
- eps: 분모를 0으로 나누는 것을 방지하기 위한 작은 상수값, 기본값은 1e-08
- betas: 긴마 값들 (beta1, beta2)로 이루어진 튜플, 기본값은 (0.9, 0.999)
- weight decay: 가중치 감쇠(L2 정규화) 계수, 기본값은 0
- amsgrad: AMSGrad 알고리즘을 사용할지 여부, 기본값은 False

임소정

https://github.com/ChanwonJung/ROKEY_team6_study/blob/main/SojeongLim/0327study_daycon(1).ipynb

```
class SimpleCNN(nn.Module):
    def __init__(self, num_classes=10):
        super(SimpleCNN, self).__init__()
        self.conv1 = nn.Conv2d(1, 32, kernel_size=3, padding=1)
        self.conv2 = nn.Conv2d(32, 64, kernel_size=3, padding=1)
        self.fc1 = nn.Linear(64 * 8 * 8, 128)  # 출력 크기 수정
        self.fc2 = nn.Linear(128, num_classes)

    def forward(self, x):
        x = torch.relu(self.conv1(x))
        x = torch.max_pool2d(x, 2)  # (32x32 -> 16x16)
        x = torch.relu(self.conv2(x))
        x = torch.max_pool2d(x, 2)  # (16x16 -> 8x8)
        x = x.view(x.size(0), -1)  # 배치 크기를 제외하고 나머지 차원 펼침
        x = torch.relu(self.fc1(x))
        x = self.fc2(x)
        return x
```

활동평가	주어진 train data를 분석하고 특징을 추출한 머신러닝 모델을 만들어 test set에 적용함.
과제	4/7 정기 평가 대비를 위한 수업 내용 복습.
향후 계획	4/3 (목) 21시: 정기 평가 대비 수업 리뷰 -AI개론/응용 편에서 공부한 챕터 하나 골라 발표 예정.
첨부 자료	 https://github.com/ChanwonJung/ROKEY_team6_study/tree/main