딥러닝 모델을 활용한
흉부 CT 이미지 분류

20202659 이효빈
20202665 천예은
20202667 최여진

목차

1. 중간 발표 Review
2. 데이터 설명
3. 모델링
4. 결과

1. 중간 발표 Review

문제 정의

- 의료 영상 진단 분야 등의 **의료 인공지능의 발전**

  → CNN 기반의 **딥러닝 모델**을 활용한

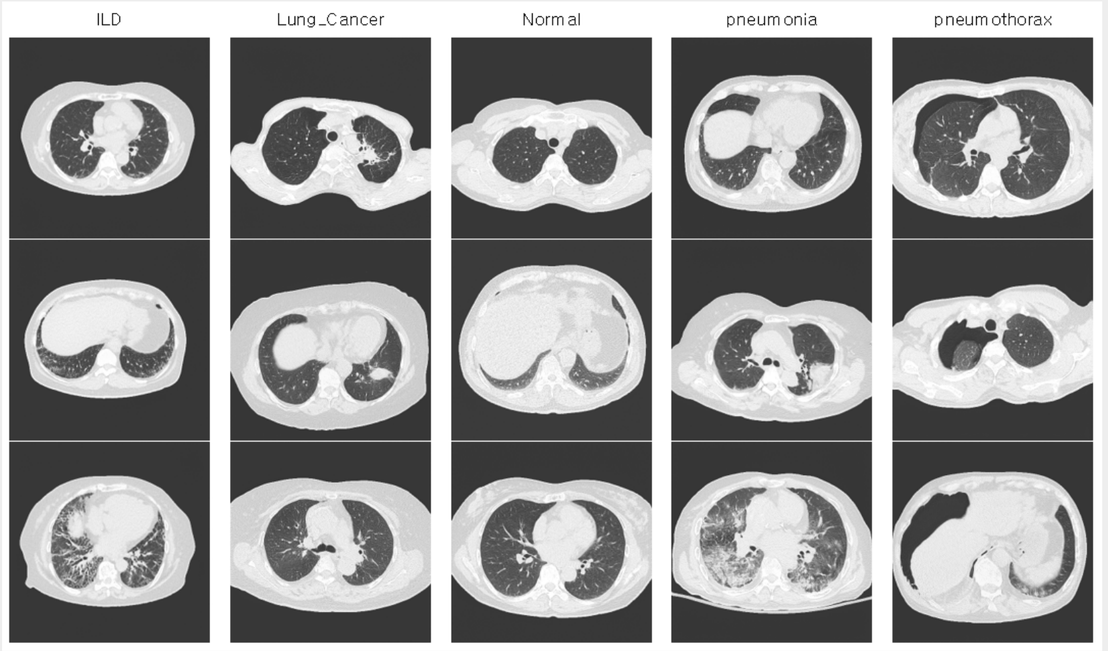
  흉부 CT 영상 이미지의 정상 및 질병 종류 분류

2. 데이터

데이터 소개

- 어반데이터톤 "Medical Synthetic Data AI Challenge"
  [Track2] 흉부 CT 데이터를 이용한 질병 종류 판별 공모전 데이터 활용

1)



1) "*어반데이터톤*," 2022년 10월 29일 검색, [Task 2] 흉부 CT 데이터를 이용한 질병 종류 판별, http://urbandatathon.com/hackathon/scheduleDetail/1001.

2. 데이터

분류 class

- 다중 분류
  - → ILD (0)
  - → LUNG_Cancer (1)
  - → Normal (2)
  - → pneumonia (3)
  - → pneumothorax (4)

```
In [9]: print(train_set.classes)
        print(val_set.classes)

['ILD', 'Lung_Cancer', 'Normal', 'pneumonia', 'pneumothorax']
['ILD', 'Lung_Cancer', 'Normal', 'pneumonia', 'pneumothorax']
```

2. 데이터

데이터

- 흉부 CT 영상 이미지

  → Train : 8001 개 / Validation : 1000 개

  → Label : 0 ~ 4

  → Size : 512 * 512

	ILD	LUNG_Cancer	Normal	pneumonia	pneumothorax
Train	1000	1000	4001	1000	1000
Validaton	125	125	500	125	125

2. 데이터

데이터

- 이미지를 Tensor로 변환

```
trans = transforms.Compose([transforms.ToTensor(),
                            transforms.Normalize(mean=[0.485, 0.456, 0.406],
                                                 std=[0.229, 0.224, 0.225])])
```

3. 모델링

성능 평가 지표

```
# 성능 평가 지표
from sklearn.metrics import f1_score
f1score = f1_score(labels.data.cpu(), preds.data.cpu(), average = 'macro')
```

- f1_score (average = 'macro')
  - → 각 class에 대한 precision의 합 / class의 개수
  - → 모든 class의 값에 동일한 가중치로 평균.
  - → class 불균형 데이터에 대해 효과적.

3. 모델링

모델

- pretrained model 활용
  - → VGG
  - → AlexNet
  - → ResNet
  - → DenseNet
  - → MobileNetV3

3. 모델링

모델 학습 및 평가 함수 정의

```python
def train_model(model, criterion, optimizer, scheduler, num_epochs=10):

    for epoch in range(num_epochs):
        print(f'Epoch {epoch}/{num_epochs - 1}')
        print('-' * 10)

        for phase in ['train', 'val']:
            if phase == 'train':
                model.train()  # 모델을 학습 모드로 설정
            else:
                model.eval()   # 모델을 평가 모드로 설정
```

3. 모델링

모델 학습 및 평가 함수 정의

```
# 매개변수 경사도를 0으로 설정
optimizer.zero_grad()

if phase == 'train':
    with torch.set_grad_enabled(phase == 'train'):
        outputs = model(inputs)
        _, preds = torch.max(outputs, dim=1)
        #_, preds = torch.max(outputs, 1)
        loss = criterion(outputs, labels)
        f1score = f1_score(labels.data.cpu(), preds.data.cpu(), average = 'macro')

        # 학습 단계인 경우 역전파 + 최적화
        loss.requires_grad_(True)
        loss.backward()
        optimizer.step()

if phase == 'val':
    with torch.no_grad():
        outputs = model(inputs)
        _, preds = torch.max(outputs, dim=1)
        #_, preds = torch.max(outputs, 1)
        loss = criterion(outputs, labels)
        f1score = f1_score(labels.data.cpu(), preds.data.cpu(), average = 'macro')
```

3. 모델링

VGG16 & AlexNet

```
model_conv = torchvision.models.vgg16(pretrained = True)

# 새로 생성된 모듈의 매개변수는 기본값이 requires_grad=True
for param in model_conv.parameters():
    param.requires_grad = False

num_ftrs = model_conv.classifier[6].in_features
model_conv.classifier[6] = nn.Linear(num_ftrs, 5)
```

ResNet

```
model_conv = torchvision.models.resnet152(pretrained=True)

# 새로 생성된 모듈의 매개변수는 기본값이 requires_grad=True
for param in model_conv.parameters():
    param.requires_grad = False

num_ftrs = model_conv.fc.in_features
model_conv.fc = nn.Linear(num_ftrs, 5)
```

3. 모델링

DenseNet

```
model_conv = torchvision.models.densenet161(pretrained=True)

# 새로 생성된 모듈의 매개변수는 기본값이 requires_grad=True
for param in model_conv.parameters():
    param.requires_grad = False

num_ftrs = model_conv.classifier.in_features
model_conv.classifier = nn.Linear(num_ftrs, 5)
```

MobileNetV3

```
model_conv = torchvision.models.mobilenet_v3_large(input_size = (512,512,3), pooling = 'avg',
                                                   pretrained = True, weights = 'imagenet')

# 새로 생성된 모듈의 매개변수는 기본값이 requires_grad=True
for param in model_conv.parameters():
    param.requires_grad = False

num_ftrs = model_conv.classifier[3].in_features
model_conv.classifier[3] = nn.Linear(num_ftrs, 5)
```

3. 모델링

Optimizer

- Adam

  → 각 모델들의 파라미터

  → lr = 0.001로 설정

lr_scheduler

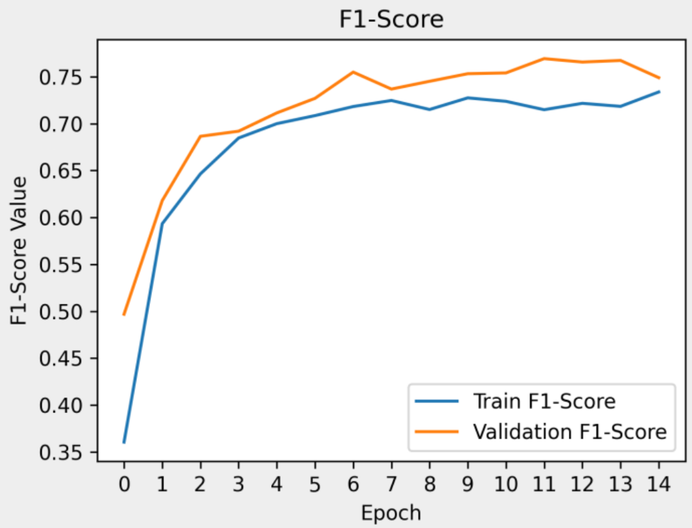
- StepLR

  → step_size = 7, gamma = 0.1로 설정

4. 결과

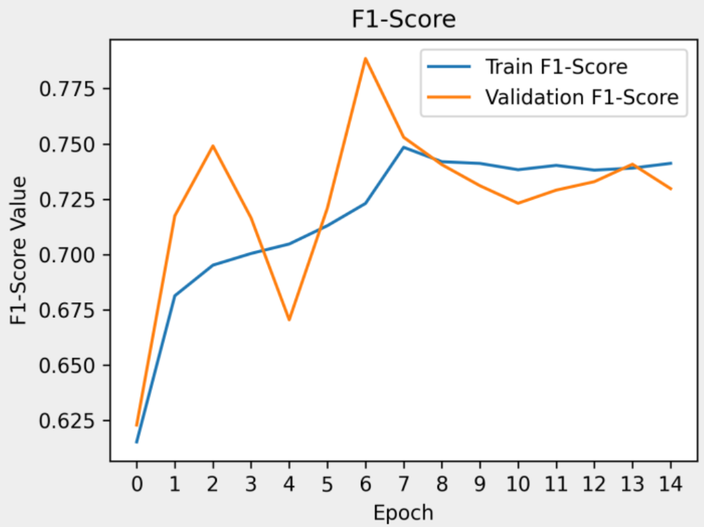
VGG16

- Best F1-Score : 0.769312



AlexNet

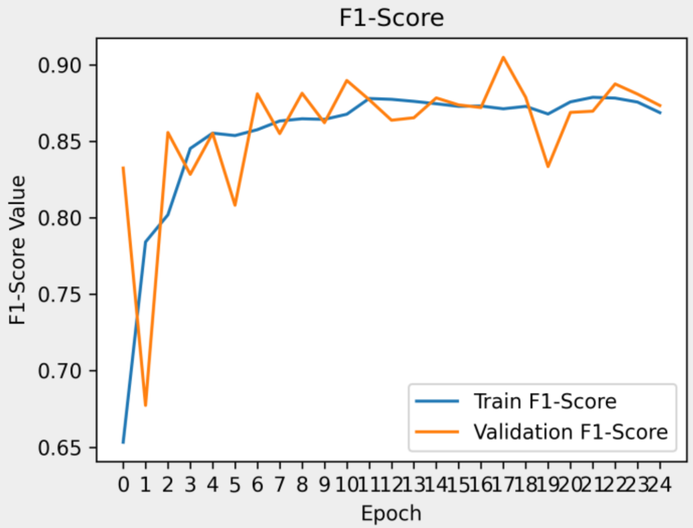
- Best F1-Score : 0.788558



4. 결과

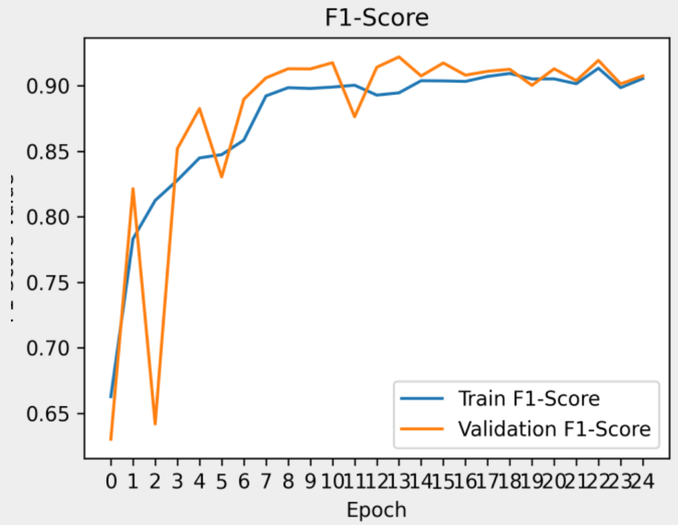
ResNet50

- Best F1-Score : 0.905039



ResNet152

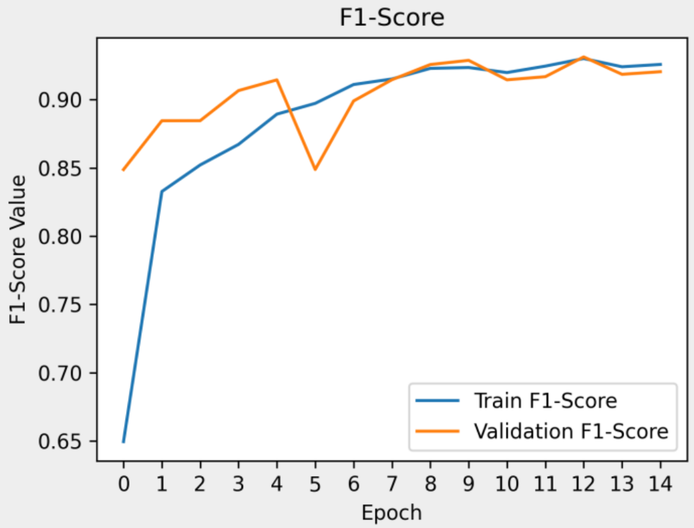
- Best F1-Score : 0.922059



4. 결과

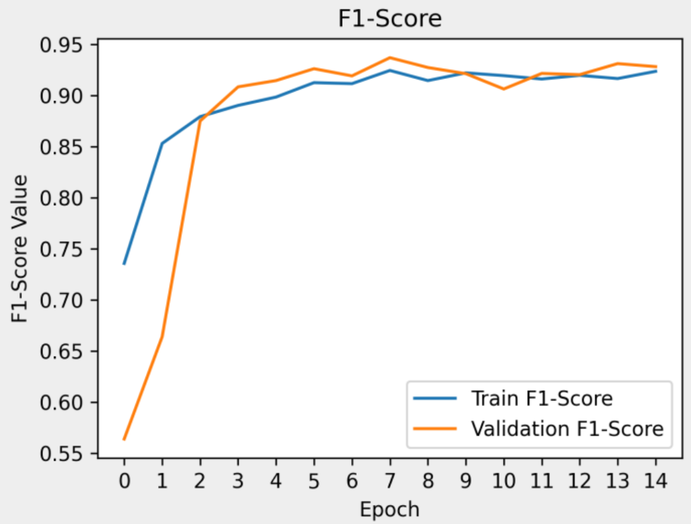
DenseNet161

- Best F1-Score : 0.931303



MobileNetV3

- Best F1-Score : 0.936962



4. 결과

모델 결과 요약

- VGG16과 AlexNet

  → 0.7대의 F1-Score로 다른 모델들에 비해 좋은 성능 보이지 못함.

- ResNet

  → 0.9대의 F1-Score로 모델의 layer가 깊어질수록 좋은 성능을 보임.

- DenseNet161과 MobileNetV3

  → 0.93대의 F1-Score로 가장 좋은 성능을 보임.

4. 결과

향후 연구 방향성

- **데이터 증대** 방법 활용

  → 이미지 회전, 이미지 크기 확대 및 축소[2)]
- RMS Prop 등의 **다양한 Optimizer 활용**
- epoch 증가
- **모델 앙상블**[3)]

2) 김민정, 김정훈, "흉부 X-선 영상에서 심장비대증 분류를 위한 합성곱 신경망 모델 제안," 한국방사선학회 논문지 15 no.5 (2021): 613-620. https://doi.org/10.7742/jksr.2021.15.5.613.
3) 허지혜 외, "전이학습기반 앙상블 딥러닝을 이용한 COVID-19 환자 영상 분류," 한국데이터정보과학회지 32 no.6 (2021): 1219-1235, DOI:10.7465/jkdi.2021.32.6.1219.

감사합니다

참고문헌

- 김민정, 김정훈. "흉부 X-선 영상에서 심장비대증 분류를 위한 합성곱 신경망 모델 제안." 한국방사선학회 논문지 15 no.5 (2021): 613-620. https://doi.org/10.7742/jksr.2021.15.5.613.

- 허지혜, 이수빈, 양원혁, 임동훈. "전이학습기반 앙상블 딥러닝을 이용한 COVID-19 환자 영상 분류." 한국데이터정보과학회지 32 no.6 (2021): 1219-1235. DOI:10.7465/jkdi.2021.32.6.1219.

- "어반데이터톤," 2022년 10월 29일 검색, [Task 2] 흉부 CT 데이터를 이용한 질병 종류 판별, http://urbandatathon.com/hackathon/scheduleDetail/1001.