2022 전기 졸업과제 중간 보고서

제조영상 불량 검출 및 분류 인공지능 학습모델 개발

A 스팀

지도교수 감진규

조장 201624495 서지훈

조원 201624426 김무영

2022.08

목차

1. 요구조건 및 제약 사항 분석에 대한 수정사항

1.1 요구 조건

철강의 표면에서 결함을 검출하여 종류를 분류할 수 있는 모델을 개발한다.

- 모델에 적합하게 데이터를 전처리를 수행
- 전처리된 데이터의 특징을 추출하여 학습하는 모델 구현
- 학습된 모델에서 데이터의 결함을 검출 및 종류 분류

1.2 제약사항 분석 및 수정사항

데이터 사용	기존에는 두 데이터를 모두 전처리한 후 모델을 학습시키기로 하였으나, 데이터가 제공된 결함의 위치를 처리하는 것에 난항을 겪어 우선 이미지를 바로 학습시키는 형태로 사용하였음.
모델 구현	본래는 tensorflow.keras 에서 제공되는 모델을 활용할 계획이었으나, 대신 간단한 모델에서부터 직접 구현하여 차후 수정해 나가는 것으로 계획을 변경하였음.

2. 설계 상세화 및 변경 내역

2.1 데이터 전처리

본래는 xml 과 csv 의 파싱을 거쳐 데이터를 전처리 할 예정이었으나 어려움을 겪어 우선 이미지를 기반으로 하는 모델을 설계하기로 하였다.

2.2 모델 설계

tensorflow.keras 에서 제공되는 여러 모델을 사용하는 대신, 간단한 모델부터 구현한 후 차후 모델을 직접 수정하기로 하였다.

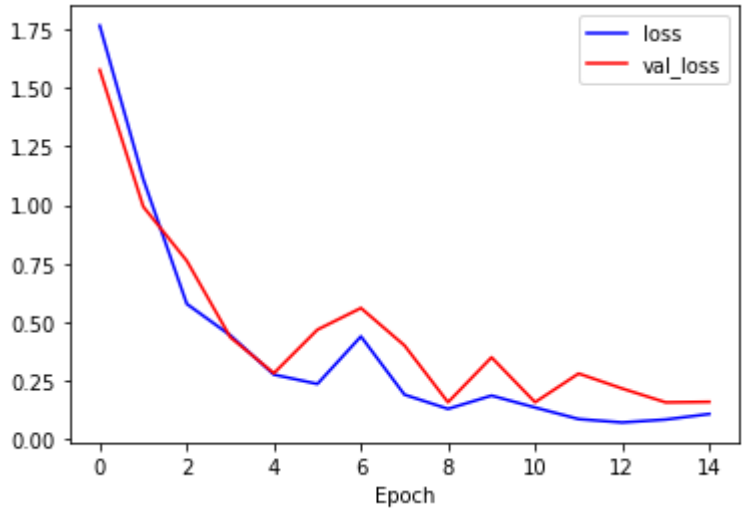
이에 따라 세 계층의 Convolutional layer, Pooling layer 와 두 개의 Dense layer 를 가진 간단한 CNN 을 초기 설계하였다.

```
model = tf.keras.Sequential()
model.add(tf.keras.layers.InputLayer(input_shape = (200, 200, 3)))

model.add(tf.keras.layers.Conv2D(32, 3, activation='relu'))
model.add(tf.keras.layers.MaxPooling2D())

model.add(tf.keras.layers.Conv2D(32, 3, activation='relu'))
model.add(tf.keras.layers.MaxPooling2D())

model.add(tf.keras.layers.Conv2D(32, 3, activation='relu'))
model.add(tf.keras.layers.MaxPooling2D())

model.add(tf.keras.layers.Flatten())

model.add(tf.keras.layers.Dense(128, activation='relu'))
model.add(tf.keras.layers.Dense(6, activation='softmax'))

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

2.3 모델 평가

```
result = model.evaluate(NEU_test_dataset)
```

```
12/12 [==============================] - 0s 14ms/step - loss: 0.2020 -
accuracy: 0.9417
```



간단한 모델이고, 데이터 또한 작기 때문에 정확도는 별로 높지 않다. 차후 더 많은 데이터를 가진 Severstal 데이터셋을 활용하여 추가로 학습을 진행하고, 모델의 Convolutional layer 를 추가하는 등으로 정확도를 높일 수 있을 것이다.

3. 갱신된 과제 추진 계획

5 월 1 주 ~ 5 월 3 주	1 번 데이터 전처리
5 월 4 주 ~ 6 월 4 주	기본 모델 작성
7 월 1 주 ~ 7 월 2 주	기본 모델 최적화
7 월 3 주 ~ 7 월 4 주	중간 보고서 작성
8 월 1 주 ~ 8 월 1 주	2 번 데이터 전처리
8 월 2 주 ~ 8 월 4 주	통합 모델 작성
9 월 1 주 ~ 9 월 2 주	통합 모델 최적화
9 월 3 주 ~ 9 월 4 주	최종 보고서 작성

4. 구성원 별 진척도

서지훈	(공통)보고서 작성 철강 결함 검출 모델 초기 형태 개발 모델 학습 파라미터 값 조정
김무영	(공통)보고서 작성 훈련용 데이터 전처리 모델 결과 분석 및 재 학습 판단

5. 보고 시점까지의 과제 수행 내용 및 중간 결과

5.1 데이터 전처리

이미지의 원활한 처리를 위하여 PIL 라이브러리로 이미지를 로드하고, numpy 라이브러리를 이용하여 배열로 전환한다.

```
NEU_train_base_path = 'NEU-DET/train/images/'
NEU_validation_base_path = 'NEU-DET/validation/images/'

NEU_datas = []
NEU_val_data, NEU_val_label = [], []

NEU_folder_paths = ["crazing/crazing_", "inclusion/inclusion_",
"patches/patches_", "pitted_surface/pitted_surface_",
               "rolled-in_scale/rolled-in_scale_", "scratches/scratches_"]
NEU_dic = {"crazing/crazing_":0, "inclusion/inclusion_":1,
"patches/patches_":2, "pitted_surface/pitted_surface_":3,
               "rolled-in_scale/rolled-in_scale_":4,
"scratches/scratches_":5}

for folder_path in NEU_folder_paths:
    temp = []
    for i in range(1, 241):
        path = NEU_train_base_path + folder_path + str(i) + ".jpg"
        if os.path.isfile(path):
            temp.append(np.array(Image.open(path)))
    NEU_datas.append(temp)

for folder_path in NEU_folder_paths:
    for i in range(241, 301):
        path = NEU_validation_base_path + folder_path + str(i) + ".jpg"
        if os.path.isfile(path):
            NEU_val_data.append(np.array(Image.open(path)))
            NEU_val_label.append(NEU_dic[folder_path])
```

train, validation, test set 으로 사용하기 위하여 데이터를 6:2:2 로 랜덤 추출하여 분류한다.

```
NEU_train_data, NEU_train_label, NEU_test_data, NEU_test_label = [], [], [],
[]

for label in range(6):
    random_list = []
    while(len(random_list) < 60):
        num = random.randrange(0, 240)
        if num not in random_list:
            random_list.append(num)

    for num in random_list:
        NEU_test_data.append(NEU_datas[label][num])
        NEU_test_label.append(label)

    for i in range(240):
        if i not in random_list:
            NEU_train_data.append(NEU_datas[label][i])
            NEU_train_label.append(label)
```

데이터셋을 numpy 배열로 변환한 후, 값을 정규화한다.

```
NEU_train_data = np.array(NEU_train_data) # list to numpy
NEU_val_data = np.array(NEU_val_data)
NEU_test_data = np.array(NEU_test_data)

NEU_train_label = tf.keras.utils.to_categorical(NEU_train_label) # one-hot-
coding 으로 변환
NEU_val_label = tf.keras.utils.to_categorical(NEU_val_label)
NEU_test_label = tf.keras.utils.to_categorical(NEU_test_label)

NEU_train_data = NEU_train_data.astype(np.float32) / 255.
NEU_val_data = NEU_val_data.astype(np.float32) / 255.
NEU_test_data = NEU_test_data.astype(np.float32) / 255.
```

데이터를 학습에 사용하기 위하여 데이터셋으로 변환한다.

```
NEU_train_dataset = tf.data.Dataset.from_tensor_slices((NEU_train_data,
NEU_train_label)).shuffle(buffer_size=1080).batch(30).repeat()

NEU_val_dataset = tf.data.Dataset.from_tensor_slices((NEU_val_data,
NEU_val_label)).shuffle(buffer_size=360).batch(30).repeat()

NEU_test_dataset = tf.data.Dataset.from_tensor_slices((NEU_test_data,
NEU_test_label)).shuffle(buffer_size=360).batch(30)
```

5.2 모델 설계

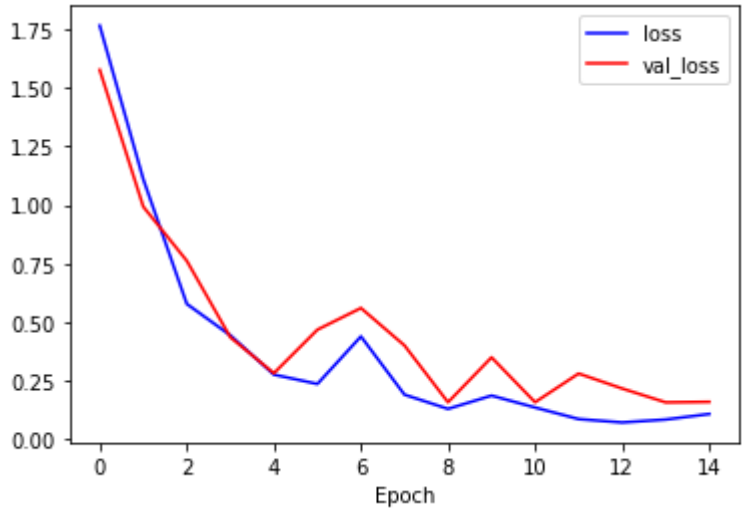
3 개의 Convolution layer, Pooling Layer 와 두 개의 Dense Layer 를 사용하여 초기 모델을 설계하였다.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 198, 198, 32)      896

 max_pooling2d (MaxPooling2D  (None, 99, 99, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 97, 97, 32)        9248

 max_pooling2d_1 (MaxPooling  (None, 48, 48, 32)       0
 2D)

 conv2d_2 (Conv2D)           (None, 46, 46, 32)        9248

 max_pooling2d_2 (MaxPooling  (None, 23, 23, 32)       0
 2D)

 flatten (Flatten)           (None, 16928)             0

 dense (Dense)               (None, 128)               2166912

 dense_1 (Dense)             (None, 6)                 774

=================================================================
Total params: 2,187,078
Trainable params: 2,187,078
Non-trainable params: 0
_________________________________________________________________
```

이는 초기 모델으로, Severstal 데이터셋의 학습이 완료된 후,

5.3 학습 결과

학습된 모델을 test set 으로 검증한 결과는 다음과 같다.



validation loss 가 줄어드는 경향은 보이나, 값이 급격하게 변화되는 구간이 존재한다. 이는 모델에 사용된 NEU-DET 데이터셋의 각각의 결함의 크기가 180 장의 이미지로, 충분한 학습에 사용되기에는 작기 때문으로 추정된다.

Severstal 의 데이터를 학습한 후의 모델도 이런 현상이 지속된다면, 데이터가 작은 NEU-DET 데이터셋의 사용을 포기해야 할 수도 있다고 판단된다.