지능화 캡스톤 프로젝트

프로젝트 #1 결과 발표

2022. 4. 13

충북대학교 산업인공지능학과

[21-7조] 봉은정, 김원우

INDUSTRIAL AI RESEARCH CENTER

수행방법 및 기여도

수행방법

- 각 팀원이 자유롭게 전체 프로젝트 구현

- 데이터 증량, 모델 구성, 코딩, 학습 결과 등 구현 방법 공유

- 학습 환경 및 결과 비교

업무분장 및 기여도

이름	비중	수행내용	비고
봉은정	50%	• 데이터 증량 / 코딩 / 학습 • 발표자료 작성 및 수정 • 발표	
김원우	50%	• 데이터 증량 / 코딩 / 학습 • 발표자료 수정 • 발표	

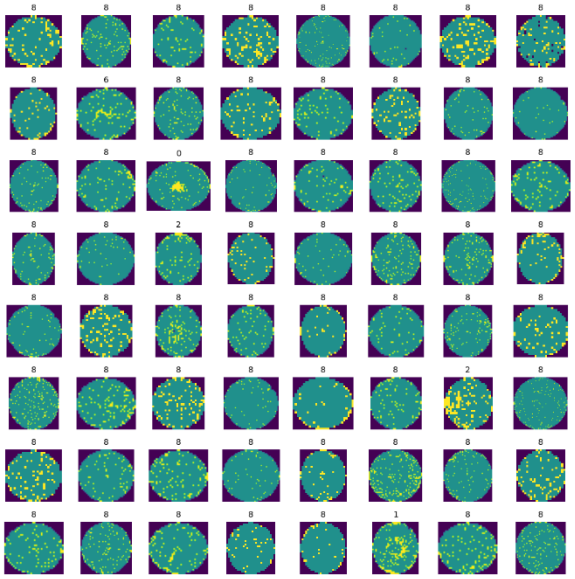
데이터셋

Dataset

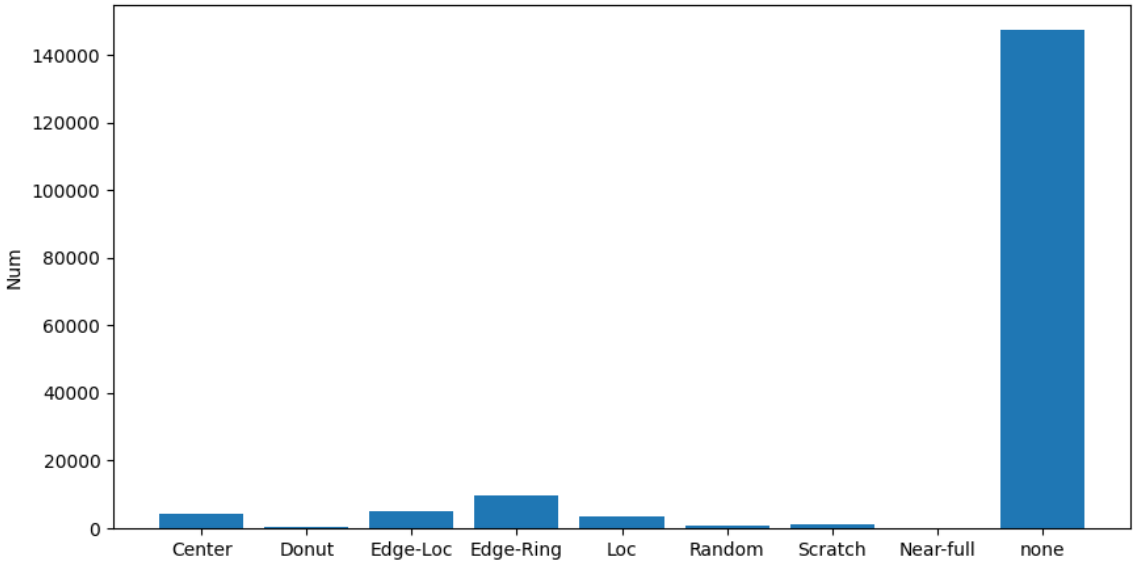
- Kaggle에서 제공하는 WM-811K wafer map Dataset 활용

- 총 811,457개의 WM Data로 구성

- Class 불균형 문제



Wafer Map Dataset



Class 분포

데이터셋

Data 전처리 및 구성

- Without Label Data 제외

- Data augmentation을 적용하여 class 불균형 해결

- Dataset 구성 → Train : Validation : Test = 65 : 20 : 15

With Label
172,950 (21.3%)

Center	4,294 (0.53%)
Donut	555 (0.07%)
Edge-Loc	5,189 (0.64%)
Edge-Ring	9,680 (1.19%)
Local	3,593 (0.04%)
Random	866 (0.11%)
Scratch	1,193 (0.15%)
Near-Full	149 (0.02%)
None	147,431 (18.17%)

기존 Dataset

Center	10,000
Donut	10,000
Edge-Loc	10,000
Edge-Ring	10,000
Local	10,000
Random	10,000
Scratch	10,000
Near-Full	10,000
None	10,000

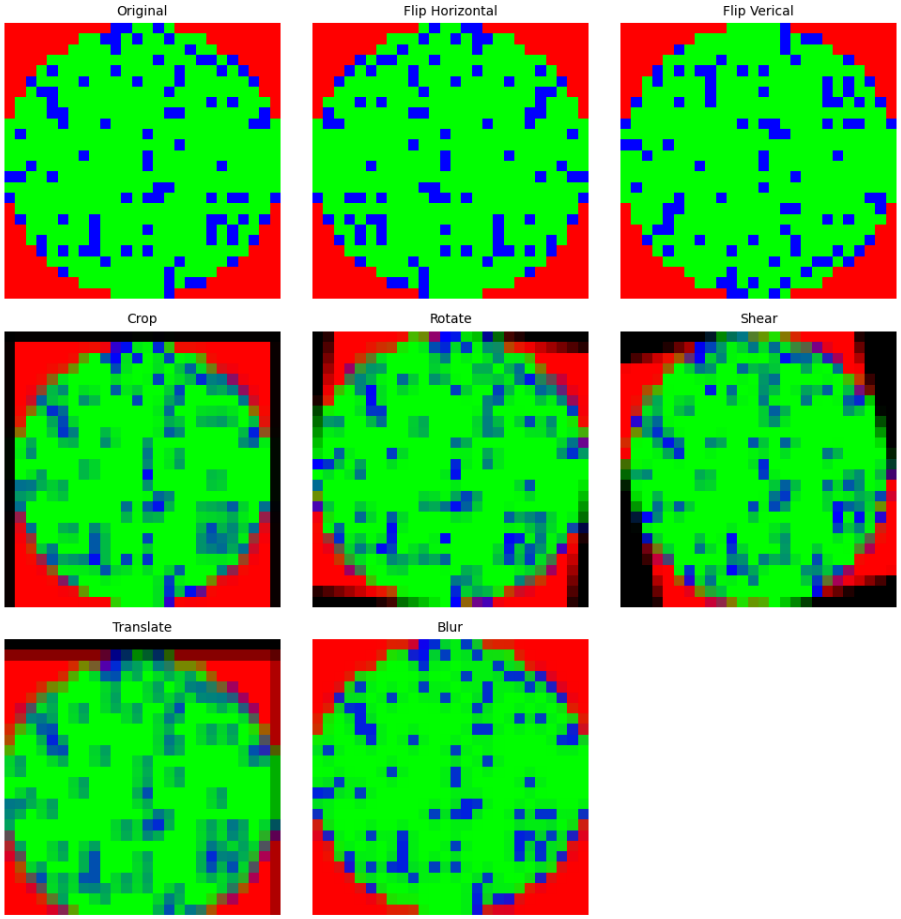
최종 Dataset

데이터셋

Data Augmentation

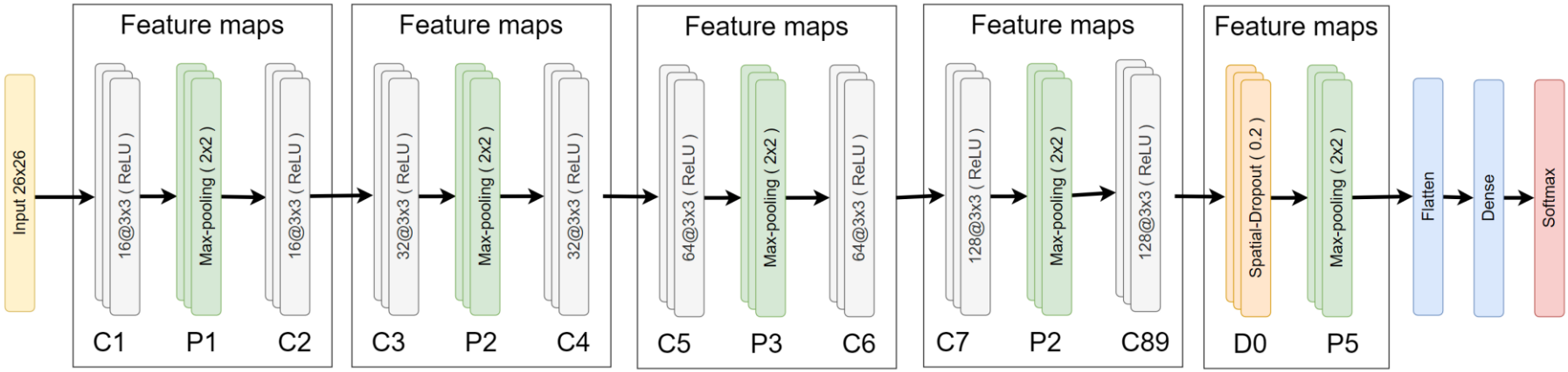
- imgaug 라이브러리 이용 (Flip, Crop, Rotate, Shear, Translate, Blur)

- 각 Class 당 10,000장으로 Dataset 균형화



CNN 구조

CNN 구조



Layer	Type	Feature Maps	Output Size	Filter Size	Padding	Activation
In	Input	3 (RGB)	26x26	-	-	-
C1	Convolution1	16	26x26	3x3	No	ReLU
P1	Max Pooling 1	16	13x13	2x2	No	-
C2	Convolution 2	16	13x13	3x3	Yes	ReLU
C3	Convolution 3	32	13x13	3x3	Yes	ReLU
P2	Max Pooling 2	32	7x7	2x2	No	-
C4	Convolution 4	32	7x7	3x3	Yes	ReLU
C5	Convolution 5	64	7x7	3x3	Yes	ReLU
P3	Max Pooling 3	64	4x4	2x2	No	-
C6	Convolution 6	64	4x4	3x3	Yes	ReLU
C7	Convolution 7	128	4x4	3x3	Yes	ReLU
P4	Max Pooling 4	128	2x2	2x2	No	-
C8	Convolution 8	128	2x2	3x3	Yes	ReLU
P5	Max Pooling 5	128	1x1	2x2	No	-
FC1	Fully-Connected 1	1	512	-	-	ReLU
FC2	Fully-Connected 2	1	128	-	-	ReLU
OUT	Output	1	9	-	-	Softmax

CNN 구조

규제화(regulation)

- CNN 모델의 과적합을 방지하기 위한 방법

1. Batch Normalization
   - 학습 속도 개선 방법

   - 가중치의 scale 정규화하여 gradient vanishing/exploding 방지

2. Dropout
   - 일반적인 모델 성능 향상 방법

   - 뉴런의 연결을 무작위로 제거하여 과적합 방지

   - 해당 프로젝트에서는 Spatial Dropout을 사용하여 20% 의 노드를 학습에서 제거

CNN 구조

주요 코드

- Keras 딥러닝 프레임워크 이용

```
def paper_model():
    input_shape = (26, 26, 3)
    input_tensor = Input(input_shape)

    # 1st feature
    conv_1 = layers.Conv2D(16, (3, 3), activation='relu', padding='same')(input_tensor)
    pool_1 = keras.layers.MaxPool2D(pool_size=(2, 2), strides=(2, 2), padding='SAME')(conv_1)
    conv_2 = layers.Conv2D(16, (3, 3), activation='relu', padding='same')(pool_1)

    # 2nd feature
    conv_3 = layers.Conv2D(32, (3, 3), activation='relu', padding='same')(conv_2)
    pool_2 = keras.layers.MaxPool2D(pool_size=(2, 2), strides=(2, 2), padding='SAME')(conv_3)
    conv_4 = layers.Conv2D(32, (3, 3), activation='relu', padding='same')(pool_2)

    # 3rd feature
    conv_5 = layers.Conv2D(64, (3, 3), activation='relu', padding='same')(conv_4)
    pool_3 = keras.layers.MaxPool2D(pool_size=(2, 2), strides=(2, 2), padding='SAME')(conv_5)
    conv_6 = layers.Conv2D(64, (3, 3), activation='relu', padding='same')(pool_3)

    # 4th feature
    conv_7 = layers.Conv2D(128, (3, 3), activation='relu', padding='same')(conv_6)
    pool_4 = keras.layers.MaxPool2D(pool_size=(2, 2), strides=(2, 2), padding='SAME')(conv_7)
    conv_8 = layers.Conv2D(128, (3, 3), activation='relu', padding='same')(pool_4)

    drop_1 = layers.SpatialDropout2D(0.2)(conv_8)
    pool_5 = keras.layers.MaxPool2D(pool_size=(2, 2), strides=(2, 2), padding='SAME')(drop_1)

    flat = layers.Flatten()(pool_5)

    dense_1 = layers.Dense(512, activation='relu')(flat)
    dense_2 = layers.Dense(128, activation='relu')(dense_1)
    output_tensor = layers.Dense(9, activation='softmax')(dense_2)

    model = models.Model(input_tensor, output_tensor)
    model.compile(optimizer='Adam',
                  loss='categorical_crossentropy',
                  metrics=['accuracy'])

    return model
```

학습 방법

딥러닝 학습 환경

CPU	Intel(R) Core(TM) i9-10900X CPU @ 3.70GHz
GPU	NVIDIA GeForce RTX 3090

하이퍼 파라미터

epoch	30
batch size	1024
learning rate	0.001
optimizer	Adam
loss function	Softmax

추가 진행 사항

- Optimizer, Learning Rate 변경 후 학습 추이 관찰

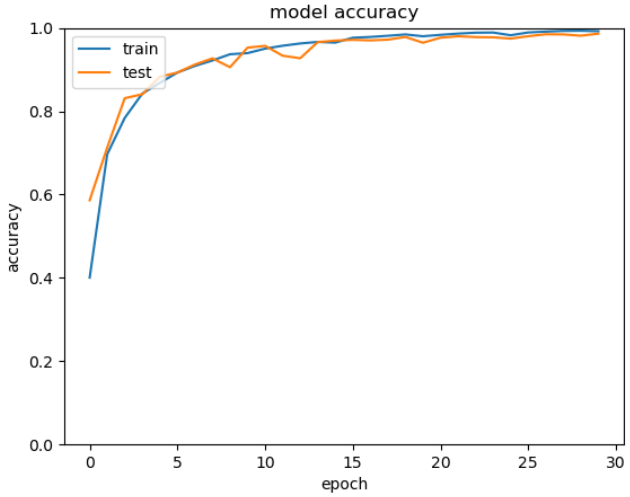
학습 방법

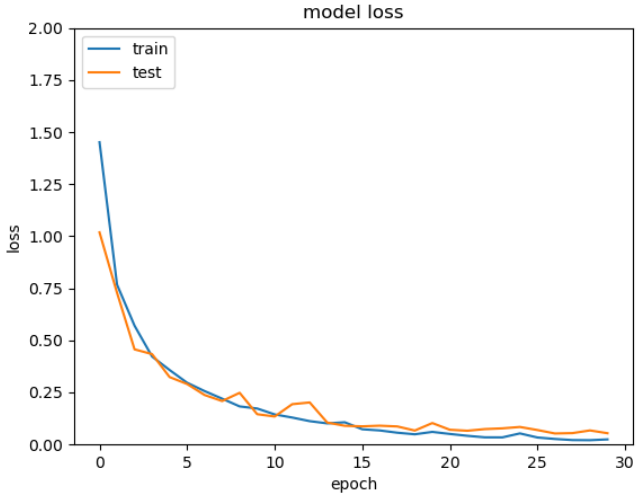
딥러닝 학습 시간

- 전체 학습 시간 → 42s

- 1 epoch 당 평균 학습 시간 → 1.4s

학습 추이 그래프

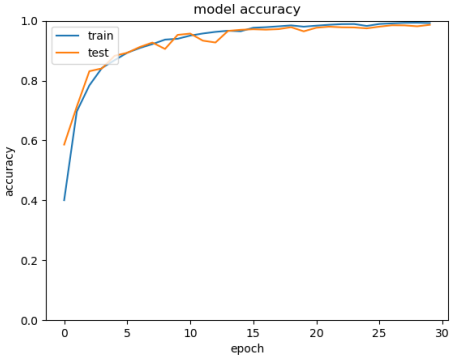




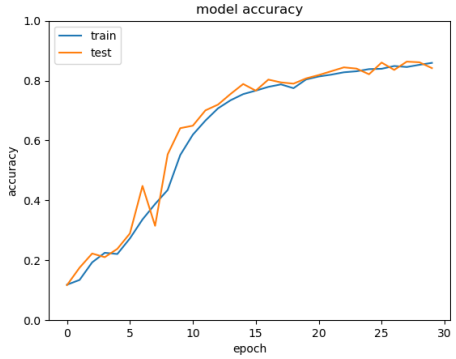
학습 추이 비교

Optimizer

- Adam Optimizer 이용 시 더 높은 성능 기록



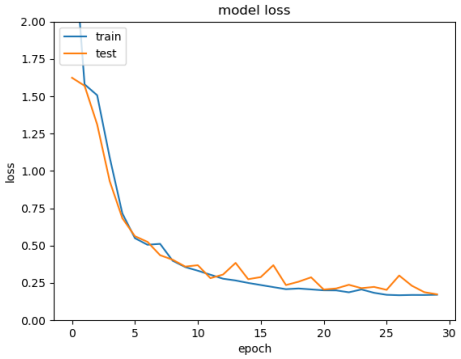
Adam



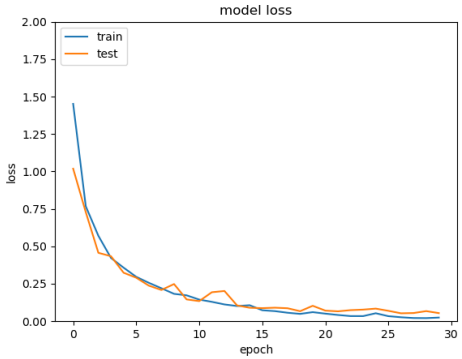
SGD

Learning Rate (Adam)

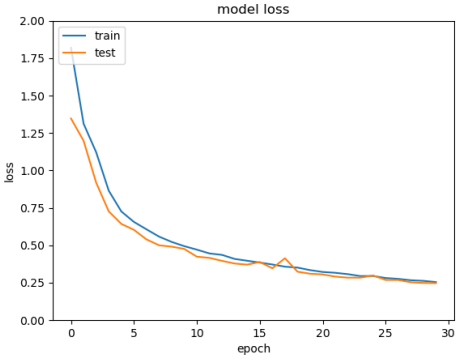
- learning rate가 적절하지 않을 경우 최저점에 수렴하지 못함



lr = 0.01



lr = 0.001

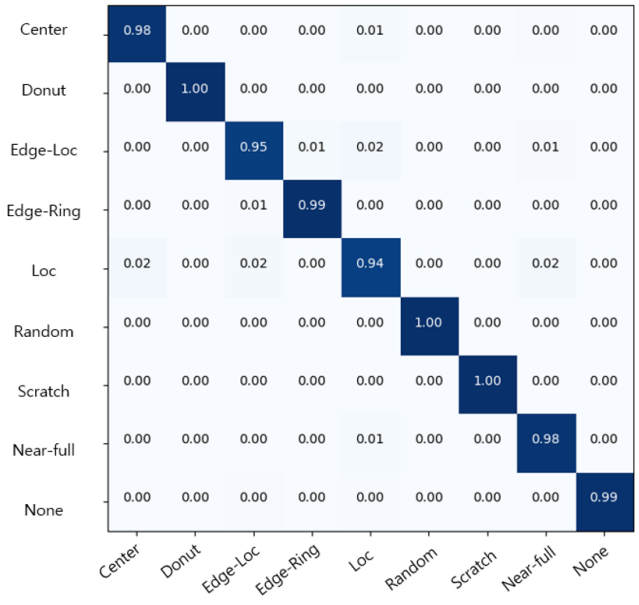


lr = 0.0001

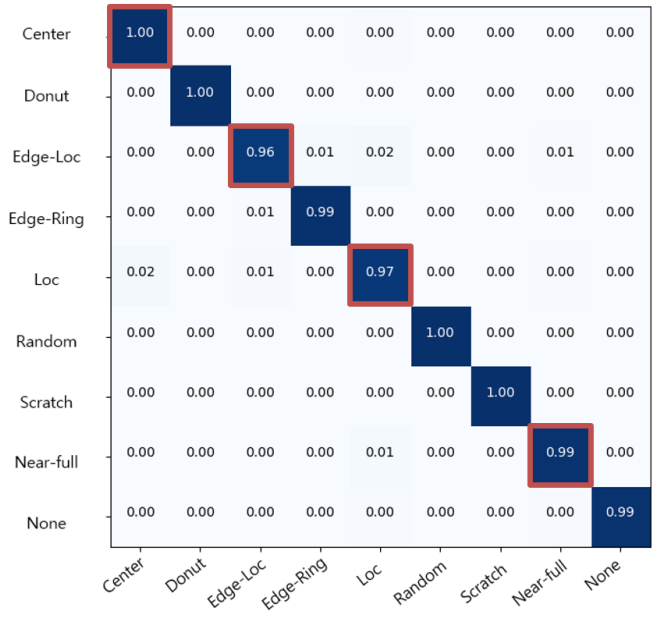
결과 및 토의

분류 성능

- Confusion matrix



Custom Model



Custom Model
(Dropout)

- 성능 비교

Classifier	Training Acc	Validation Acc	Precision	Recall	F1-Score
CNN-WDI	98.9	96.4	96.2	96.2	96.2
Custom Model	99.9	98.1	97.9	98.1	98.0
Custom Model (Dropout)	99.9	98.3	98.6	98.6	98.6

결과 및 토의

결과

- 적절한 Optimizer 설정의 중요성 확인

- Learning Rate와 모델 성능 관계 확인

- Spatial Dropout을 적용하여 성능 향상

추후 개선 및 테스트 사항

- Batch Normalization 적용

- Spatial Dropout 비율 변경 후 성능 비교

- Input 사이즈 변화 후 성능 비교

감사합니다