Technical Paper

Journal of Aerospace System Engineering
Vol.16, No.4, pp.109-115 (2022)

EISSN 2508-7150
http://dx.doi.org/10.20910/JASE.2022.16.4.109

합성곱 신경망을 이용한 손상된 볼트의 이미지 분류

이수병[1] · 이석순[2,†]
[1]경상국립대학교 기계공학부

Image Classification of Damaged Bolts using Convolution Neural Networks

Soo-Byoung Lee[1], Seok-Soon Lee[2,†]
[1]School of Mechanical Engineering, Gyeongsang National University

Abstract

The CNN (Convolution Neural Network) algorithm which combines a deep learning technique, and a computer vision technology, makes image classification feasible with the high-performance computing system. In this thesis, the CNN algorithm is applied to the classification problem, by using a typical deep learning framework of TensorFlow and machine learning techniques. The data set required for supervised learning is generated with the same type of bolts. some of which have undamaged threads, but others have damaged threads. The learning model with less quantity data showed good classification performance on detecting damage in a bolt image. Additionally, the model performance is reviewed by altering the quantity of convolution layers, or applying selectively the over and under fitting alleviation algorithm.

초 록

딥러닝 기법과 컴퓨터 비전 기술을 융합한 합성곱 신경망 알고리즘은 고성능 컴퓨팅 시스템을 기반으로 이미지 데이터의 분류를 가용하게 한다. 본 논문에서는 합성곱 신경망 알고리즘을 대표적인 딥러닝 프레임워크인 텐서플로와 학습 기법을 이용하여 구현하고 이미지 분류 문제에 적용한다. 모델의 지도학습에 필요한 데이터는 동일 종류의 볼트를 이용하여 나사산이 정상인 볼트와 나사산이 손상된 볼트로 구분하여 이미지를 생성하였다. 소량의 이미지 데이터를 이용한 학습 모델은 좋은 성능으로 볼트의 손상을 탐지하였다. 그리고 모델의 내부 구성에 따른 학습 성능을 비교하기 위해 합성곱 신경망 내 컨볼루션 레이어의 개수를 변경하고 과적합 회피기법을 선택 적용하여 이미지 분류 성능을 확인하였다.

Key Words : 이미지 분류(Image Classification), 합성곱 신경망(Convolution Neural Networks), 지도학습(Supervised Learning), 랜덤 포레스트(Random Forest), 기계학습(Machine Learning)

1. 서 론

근래 인공 신경망 기반 기술은 고성능 그래픽처리장치가 적용된 컴퓨팅 시스템과 결합하여 사람의 인지행동을 모사하는 인공지능 기능을 구현한다. 일상생활에서 광학 카메라를 이용해 차량번호를 자동으로 인식하고 분석 처리하는 컴퓨터 비전 기술을 쉽게 접할 수 있다. 이처럼 광학장비의 발전으로 확보된 양질의 데이터는 인공지능(AI) 기술과 융합하여 보안, 감시, 정찰 분야에서 높은 신뢰도의 탐지와 분류 성능을 제공한다.

본 논문에서는 합성곱 신경망 알고리즘을 이용하여 손상된 볼트의 이미지를 분류하는 문제에 적용한다. 양질의 소량 데이터를 이용한 지도학습 모델이 좋은 성능으로 분류 가능함을 확인하고 학습 모델의 내부 구성에 따른 분류 성능을 비교하였다.

Received: Mar. 28, 2022 Revised: Jun. 02, 2022 Accepted: Jul. 05, 2022
† Corresponding Author
Tel: *** - **** - **** E-mail: leess@gnu.ac.kr

인공지능 학습환경으로 하드웨어는 제온(Xeon) 프로세서 E-2144G, 메모리 16 GB RAM, 그래픽처리장치 NVIDIA Quadro P620 2 GB를 사용하고 소프트웨어는 Windows 10 Pro 64 bit 운영체제에서 통합개발환경 Spyder에 Python 3.9, Tensorflow-GPU 2.6, CUDA 11.4, cuDNN 8.2 패키지를 설치하였다. 그리고 필요시 구글 코랩(Colab) 운용환경을 사용하였다.

2. 합성곱 신경망 기초 이론

인공 신경망의 기본 단위인 퍼셉트론(Perceptron)은 단일 신경세포를 모사하여 Fig. 1과 같이 입력 x에 대해 가중치 w를 곱하고 편향 b를 더한 가중 합을 활성화 함수를 거쳐 최종 출력으로 전달하도록 구성된다.

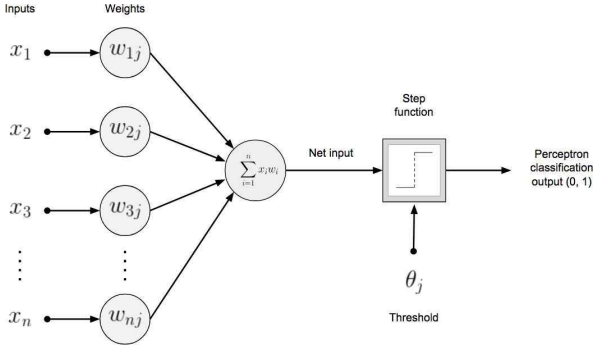

Fig. 1 Single-Layer Perceptron[1]

딥러닝 알고리즘은 Fig. 2과 같이 다층 퍼셉트론 구성을 통한 XOR 구현과 그래디언트(Gradient) 방법의 오차를 기반으로 신경망의 가중치를 수정 조절하는 역전파 알고리즘 적용 그리고 기울기 소실 문제를 해소하는 비선형 활성화 함수를 이용하여 구동한다[1].

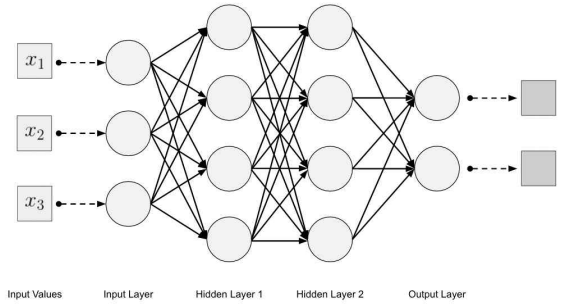

Fig. 2 Multi-Layer Neural Network[1]

합성곱 신경망은 이미지 내 동일 특징을 추출하고 처리하는 알고리즘으로 신경망의 내부는 Fig. 3과 같이 이미지 픽셀(Pixel) 간 지역적 구조를 슬라이딩 윈도우 방식으로 학습하는 컨볼루션 레이어와 기존 딥러닝 신경망의 완전연결 레이어를 결합하여 구성한다.

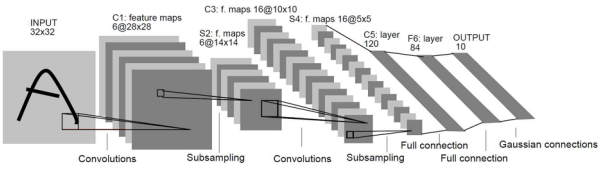

Fig. 3 CNN Architecture LeNet-5[2]

2.1 데이터

데이터의 품질, 개수, 구조는 합성곱 신경망 모델의 지도학습에 미치는 영향이 크다. 데이터셋(Data set)의 구조는 모델을 훈련하는 학습용 데이터와 손실함수를 통한 모델 성능을 확인하는 검증용 데이터 그리고 모델의 최종 성능을 평가하는 시험용 데이터로 구분한다. 시험용 데이터는 학습과 검증에 사용되지 않은 별도의 데이터로 구분한다.

2.2 컨볼루션 레이어

컨볼루션(Convolution) 레이어는 Fig. 4와 같이 전체 이미지에서 탐색 커널(Kernel)을 일정한 스트라이드(Stride)로 이동하는 방식으로 이미지의 특징을 찾는다. 컨볼루션 연산 과정에서 이미지 외곽이 한 픽셀씩 줄어드는 현상이 발생하는데 이것을 방지하기 위해 이미지 출력 형태가 입력과 동일하게 유지되는 패딩(Padding) 기능을 설정하여 영상의 크기를 한 픽셀씩 크게 하고 0 또는 평균값을 적용한다[1].

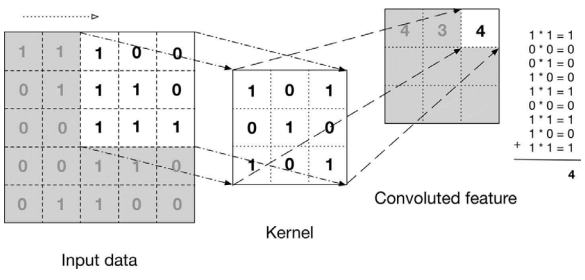

Fig. 4 Convolution Operation[1]

컨볼루션의 출력은 활성화 함수로 전달되며 활성화 함수는 기울기 소실 문제가 발생하는 Sigmoid 또는 Tanh 함수와 달리 입력값이 음수일 때 0을 출력하고 양수일 때 입력된 값을 그대로 출력하는 ReLu 함수를 일반적으로 적용한다[1].

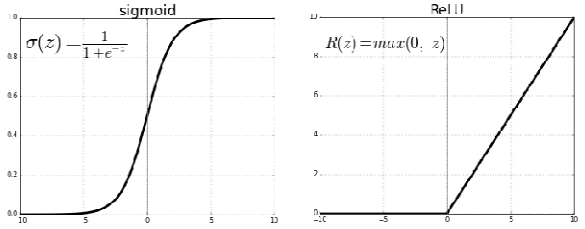

Fig. 5 Sigmoid And ReLu Function[3]

2.3 풀링 레이어

풀링(Pooling)은 이미지를 가로, 세로 방향으로 절반으로 줄이는 연산이다. 합성곱 신경망에서 주로 사용되는 맥스풀링(Maxpooling) 레이어는 영역 내에서 최대 픽셀값을 취하는 방식으로 이미지 특징을 추출하면서 이미지 크기를 줄여 연산량을 크게 줄일 수 있다.

2.4 드롭아웃

모델은 지도학습 과정에서 데이터셋에 적합하도록 반복 학습을 통해 구성 변수를 조정한다. 한 층에 속한 노드의 수 또는 층의 수가 많아지면 학습 데이터에서는 좋은 예측 결과를 보이지만 학습 모델이 훈련 데이터에 과적합 되어 일반화되지 못하거나 데이터 특징을 포착 못 하여 시험 데이터에서는 예측률이 떨어지는 과적합 현상을 확인할 수 있다[1].

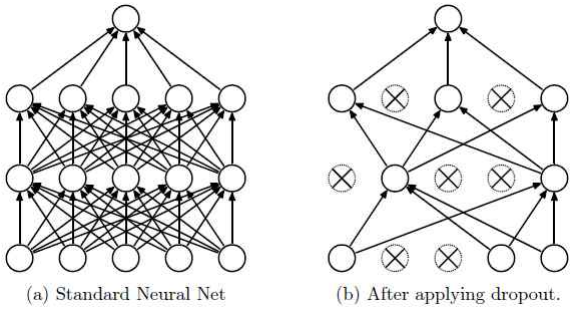

Fig. 6 Dropout[4]

드롭아웃(Dropout) 기법은 Fig. 6과 같이 모델 학습 과정에서 일시적으로 모델에서 뉴런을 제거하는 방법으로 과적합을 회피하기 위해 적용하고 검증 과정에서는 모든 뉴런 출력을 사용하여 모델 전체를 평가한다.

2.5 플래튼-덴스 레이어

플래튼(Flatten) 레이어는 2차원 배열을 1차원으로 변경한다. 그리고 덴스(Dense) 레이어는 학습 모델의 마지막 층으로 기본층에 연결한다. 그리고 활성화 함수 소프트맥스(Softmax)를 통해 출력 개수의 차원을 갖는 벡터 성분으로 입력값 사이 대비가 크고 전체 합이 1이 되는 확률적 데이터 성분을 갖게 한다.

2.6 모델 컴파일(Compile)

모델은 정의된 딥러닝 또는 머신러닝 기법을 통해 입력과 출력에 관한 신경망 학습을 수행한다.

2.6.1 옵티마이저와 손실함수

딥러닝은 스토캐스틱(Stochastic) 방법으로 그래디언트 계열의 적응형 수치 최적화 알고리즘을 적용한 옵티마이저(Optimizer)와 손실함수를 적용하여 모델을 학습한다. 배치(Batch) 크기로 나눈 전체 학습 데이터를 에포크(Epoch) 수만큼 반복 학습, 검증하여 에포크마다 정확도와 손실을 계산하여 제시한다.

2.6.2 랜덤 포레스트

랜덤 포레스트(Random forest)는 머신러닝 기법으로 데이터 수량이 작을 때 전체 데이터에서 일부 데이터의 중첩을 허용하는 부트스트래핑(Bootstrapping) 방식으로 생성한 샘플 데이터를 학습하고 여러 개의 결정트리(Decision tree)를 이용한 분류 결과를 보팅(Voting)을 통해 결정한다.

3. 본 론

3.1 데이터 생성

합성곱 신경망 모델의 지도학습에 사용할 양질의 이미지 데이터를 생성하기 위해 동일한 각도와 조명에서

볼트를 촬영하도록 카메라와 조명을 고정 설치하였다. 그리고 카메라는 삼성 NX10 기종을 사용하였고 이미지는 너비 1920 픽셀, 높이 1280 픽셀 크기의 RGB 컬러로 설정하여 촬영하였다. 이미지 데이터는 Fig. 7과 같이 동일 종류의 볼트에 대해 나사산이 정상인 이미지와 나사산이 손상된 이미지로 구분하여 각각 98개를 생성하였다. 이 중 학습용 데이터로 정상품과 나사산이 손상된 제품의 이미지를 각각 72개, 검증용 데이터로 각각 18개 그리고 시험용 데이터로 각각 8개로 구분하여 분류하였다.

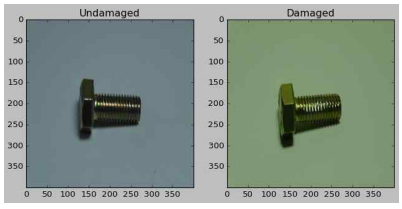

Fig. 7 Data Set Bolt Images

3.2 모델 구성

모델 학습을 위해 데이터의 배치 크기는 32로 설정하고 RGB 3채널 이미지의 입력 크기는 가로와 세로가 동일한 64 픽셀에서 800 픽셀 사이에 6단계로 구분하였다. 학습 이미지는 Fig. 8과 같이 픽셀의 밝기를 정규화 하고 데이터 학습량 부족에 따른 과적합 영향을 줄이도록 Fig. 9와 같이 입력 이미지를 같이 회전, 확대, 좌우 반전하는 데이터 증강기법을 모델 성능 검토 조건에 따라 선택적으로 적용하였다.

```
# Normalize pixel values to between 0 and 1
x_train, x_val = x_train / 255.0, x_val / 255.0

# Data Augmentation - Random Flip, Rotation, Zoom
data_augmentation = tf.keras.Sequential([
    tf.keras.layers.experimental.preprocessing.PreprocessingLayer(
        input_shape=(img_height, img_width, 3)),
    tf.keras.layers.experimental.preprocessing.RandomFlip("horizontal"),
    tf.keras.layers.experimental.preprocessing.RandomRotation(0.1),
    tf.keras.layers.experimental.preprocessing.RandomZoom(0.1)])
```

Fig. 8 Data Normalizing And Augmentation[5]

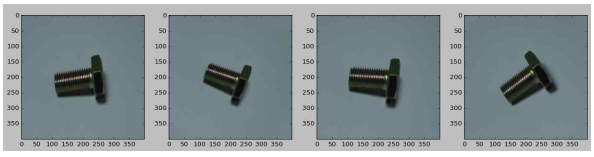

Fig. 9 Data Augmentation Bolt Images

합성곱 신경망의 기본 모델 구성은 Fig. 10과 같이 동일 패딩과 비선형 활성화 함수 ReLu를 적용한 컨볼루션 레이어와 맥스풀링 레이어를 연속하여 3회 연결하고 드롭아웃과 플래튼-덴스 레이어를 연결한다. 모델 구성을 도식화하면 Fig. 11과 같다.

```
# Model Sequential - CNN
feature_extractor = Sequential([data_augmentation,
    tf.keras.layers.Conv2D(16, 3, padding='same', activation='relu'),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Conv2D(32, 3, padding='same', activation='relu'),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Conv2D(64, 3, padding='same', activation='relu'),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Flatten()])
x = feature_extractor.output
x = Dense(128, activation='relu', kernel_initializer = 'he_uniform')(x)
prediction_layer = Dense(2, activation = 'softmax')(x)
cnn_model = Model(inputs=feature_extractor.input, outputs=prediction_layer)
```

Fig. 10 Model Sequential[5-6]

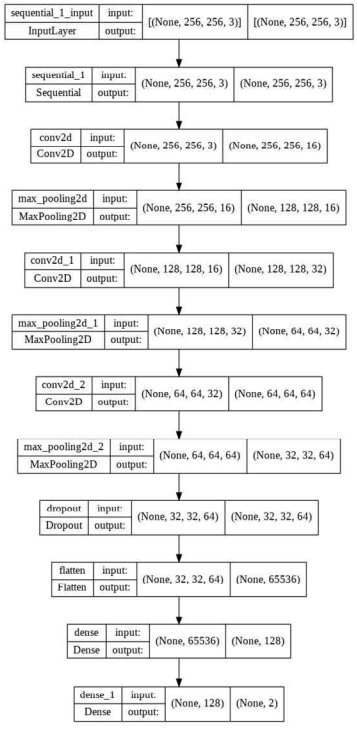

Fig. 11 Model Architecture

3.3 모델 학습

모델 학습을 위해 옵티마이저와 손실함수를 적용한 경우와 랜덤 포레스트를 결합한 경우에 하이퍼파라미터(Hyper parameter) 조정에 따른 학습 성능을 비교 확인하였다.

3.3.1 옵티마이저, 손실함수 적용

Figure 12에서와 같이 옵티마이저는 탐색 방향을 단기 누적하여 계산하고 학습률을 단기 파라미터 변화량에 반비례하게 적용하는 Adam을 사용하였고 손실함수는 Categorical_Crossentropy를 사용하였다.

```
# Compile the CNN model - Optimizer
cnn_model.compile(optimizer='Adam',
                  loss ='categorical_crossentropy', metrics = ['accuracy'])
# Train the CNN model
epochs=100
history = cnn_model.fit(x_train, y_train_one_hot, epochs=epochs,
                        validation_data = (x_val, y_val_one_hot))
```

Fig. 12 CNN Model With Optimizer Adam[6]

모델 내 컨볼루션과 맥스풀링 레이어를 단수로 구성하여 학습한 결과 Fig. 13과 같이 수렴하지 않는 경향을 보였고 복수 이상으로 구성하여 학습한 결과 Fig. 14와 같이 학습 초기에 빠르게 수렴함을 확인하였다.

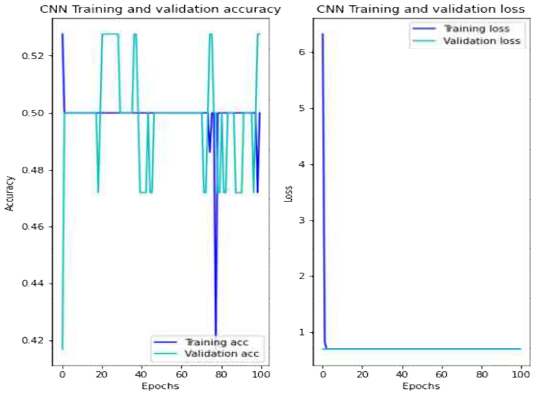

Fig. 13 One Layer Model With 256 Pixel

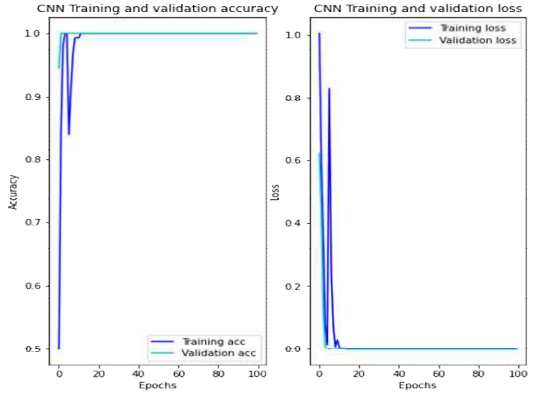

Fig. 14 Multi-Layer Model With 256 Pixel

과적합 회피기법의 효과를 확인하기 위해 컨볼루션과 맥스풀링 레이어를 3회로 적용하고 드롭아웃과 데이터 증강기법을 선택적으로 적용하였을 경우 Fig. 15와 Fig. 16과 같이 약 20회 이후 전체적으로 정확도가 향상되고 오차율이 감소하는 경향을 확인하였다.

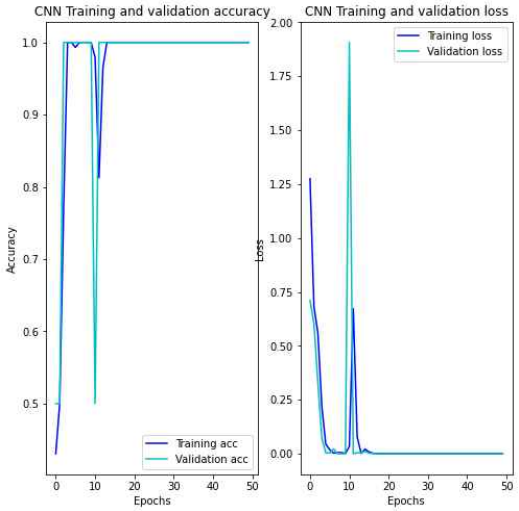

Fig. 15 Dropout With 256 Pixel

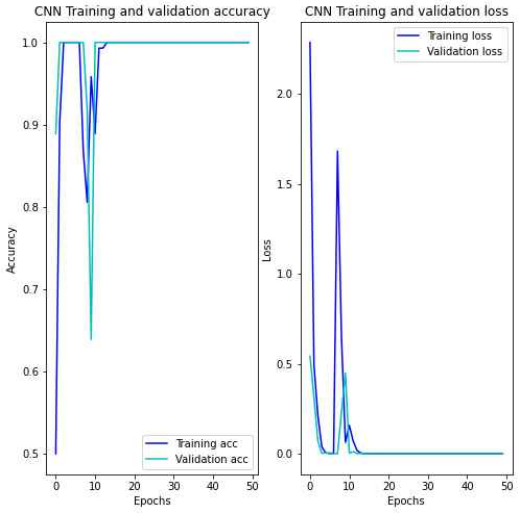

Fig. 16 Data Augmentation With 256 Pixel

3.3.2 랜덤 포레스트 적용

모델 컴파일러에 따른 학습 성능을 비교하기 위해 Fig. 17과 같이 랜덤 포레스트를 적용하였고 학습 모델의 분류 성능을 Fig. 18과 같이 확인하였다.

```
## Train the model with RandomForest
from sklearn.ensemble import RandomForestClassifier
X_train_RF = feature_extractor.predict(x_train)
rf_model = RandomForestClassifier(n_estimators = 50, random_state = 42)
rf_model.fit(X_train_RF, y_train)
# Send test data through same feature extractor process
X_val_feature = feature_extractor.predict(x_val)
# Predict using the trained RF model.
prediction_RF = rf_model.predict(X_val_feature)
# Inverse le transform to get original label back.
prediction_RF = le.inverse_transform(prediction_RF)
```

Fig. 17 CNN Model With Random Forest[6]

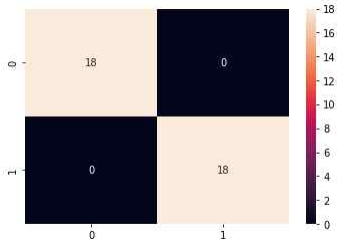

Fig. 18 Confusion Matrix

3.4 모델 성능 검토

모델 성능 검토는 모델의 내부 구성 조건에 따른 분류 성능을 학습용 이미지 크기에 따라 비교하였다. 모델 성능시험은 별도의 시험용 이미지를 이용하였다.

3.4.1 합성곱 신경망 레이어 개수 효과

모델 내 과적합 회피기법은 미적용하고 합성곱 신경망의 레이어 개수를 Table 1에 명시된 조건으로 모델을 구성하여 학습하고 시험한 결과는 Fig 19와 같다.

Table 1 Model Composition Case I

Case I	#1	#2	#3
Number of Layers	1	2	3
Dropout	X	X	X
Data Augmentation	X	X	X

X: Not Applied, O: Applied

단일 합성곱 신경망 레이어에서는 지도학습이 원활히 이루어지지 않아 분류 능력은 50% 수준으로 확인되었고 일부 이미지 크기가 큰 경우에 Adam 옵티마이저 적용 시 분류 능력이 80% 수준으로 확인되었다. 그리고 복수 이상의 합성곱 신경망 레이어와 Adam 옵티마이저 적용 시 분류 능력은 80% 이상으로 증가함을 확인하였다.

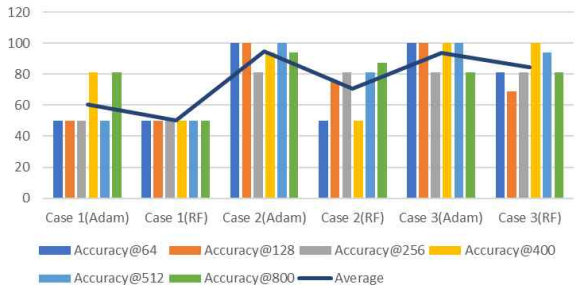

Fig. 19 Model Performance Case I

3.4.2 과적합 회피기법 효과

모델 내 합성곱 신경망 레이어는 3개를 적용하고 과적합 회피기법을 Table 2에 명시된 조건으로 모델을 구성하여 학습하고 시험한 결과는 Fig 20과 같다.

Table 2 Model Composition Case II

Case II	#1	#2	#3	#4
Number of Layers	3	3	3	3
Dropout	X	O	X	O
Data Augmentation	X	X	O	O

X: Not Applied, O: Applied

모델의 분류 성능은 전반적으로 80% 이상으로 우수한 성능을 보였다. 드롭아웃과 데이터 증강을 선택적으로 적용 시 데이터 증강기법에서 다소 높은 분류 성능 결과를 보였다. 이러한 결과는 양질이지만 소량 데이터셋을 고려할 때 학습량을 늘려서 과적합을 회피하는 데이터 증강기법이 더 유용한 것으로 고려된다. 그리고 두 가지 기법을 모두 적용하였을 때 더욱 안정적으로 정확한 분류 성능이 있음을 확인하였다.

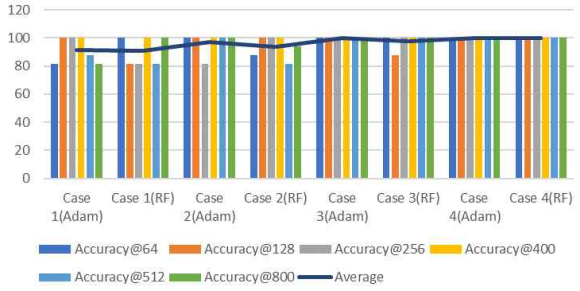

Fig. 20 Model Performance Case II

4. 결 론

본 논문에서는 합성곱 신경망에 양질의 소량 데이터 셋과 과적합 회피기법을 적용하여 볼트의 손상 여부를 탐지하는 이미지 분류 문제에 적용하였고 검토 결과 좋은 성능으로 탐지 가능함을 확인하였다.

또한, 모델의 내부 구성에 따른 학습 성능을 비교하기 위해 학습 이미지 크기에 따른 합성곱 신경망 모델에 옵티마이저와 손실함수 또는 랜덤 포레스트를 결합하여 지도학습을 수행하고 컨볼루션-맥스풀링 레이어 개수에 따른 성능과 과적합 회피기법의 선택적 적용에 따른 성능을 확인하였다. 성능시험 Case I에서 복수 이상의 합성곱 신경망 레이어를 적용하면 원활한 모델 학습이 이루어졌다. 그리고 옵티마이저 Adam과 손실함수를 이용하여 학습한 모델이 랜덤 포레스트를 적용한 모델보다 이미지 크기가 증가함에 따라 좋은 분류 성능을 보였다. 성능시험 Case II에서 3개의 합성곱 신경망 레이어가 기본 적용되어 전반적으로 좋은 분류 성능을 보였다. 과적합 회피기법을 선택적으로 적용 시 드롭아웃보다 데이터 증강기법에서 다소 더 좋은 결과를 보였다.

합성곱 신경망을 이용한 이미지 분류는 향후 광학장비의 발전을 통한 양질의 데이터 확보와 컴퓨터 비전 기술의 융합으로 다양한 문제에 적용하여 신속 정확한 결과를 얻을 수 있을 것으로 기대된다.

References

[1] Patterson, Josh, and Adam Gibson, *Deep learning: A practitioner's approach*, O'Reilly Media, 2017.

[2] LeCun, Yann, et al, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE* 86.11, pp. 2278-2324, 1998.

[3] Sagar Sharma, 2017, “Activation Functions in Neural Networks.” Towards Data Science, February 5, 2022. https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6.

[4] Srivastava, Nitish, et al, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, 15.1, pp. 1929-1958, 2014.

[5] TensorFlow (2022) Classification [Source code]. https://www.tensorflow.org/tutorials/images/classification.

[6] Sreenivas B., (2020) Classification [Source code]. https://github.com/bnsreenu/python_for_microscopists/blob/master/158_classification_CNN_RF.py.