

G.G.G 교과융합탐구대회

허예찬, 천준우, 강희전

November 3, 2020

AI를 이용해 생물 다양성 보전을 위한 식물 병충해 분류기 만들기

개요

1. 개발 동기

우리는 생물 다양성 보전을 위해 어떤 컴퓨터 프로그램이 필요할지 고민했다. 조사하던 중 생물 다양성을 보전하기 위해서는 생물들이 어떤 병에 많이 걸렸는지를 알아야 한다는 사실을 알았다.. 왜냐하면 생물이 멸종하는 원인은 치명적인 병충해이기 때문이다. 대표적인 사례는 아일랜드 대기근이다. 아일랜드 대기근의 주된 원인은 감자의 역병이었다. 이 사건으로 인해 대략 1,000,000명의 아일랜드인이 사망하였고, 1,000,000 명의 아일랜드인은 그들의 고향을 떠날 수 밖에 없었다. 이처럼 생물 다양성의 감소는 그 종 뿐만 아니라 인간과 전 생태계에 아울러 영향을 미친다. 하지만 만약 아일랜드의 감자 품종이 다양했다면 어땠을까. 그랬다면 대기근이 오지 않았을 것이다. 또한, 이런 병을 조기에 발견했다면 다른 종을 키우는 듯 생물 다양성 보전을 위한 활동과 대응을 할 수 있었을 것이다. 이처럼 생물 다양성을 보전하기 위해서는 생물이 어떤 병을 가지고 있는지 조사하는 것은 매우 중요하다.

2. 기존 방법의 문제점

이전에는 전문가가 직접 생물들을 하나하나 살펴보며 어떤 질병이 많이 있는지, 어떤 식으로 대응해야 하는지 결과를 짜야 했다. 우리는 이런 방법에서 몇 가지 비효율성을 발견했다.

<기존 조사 방법의 문제점>

1. 고급인력인 전문가가 요구된다.
2. 수작업인 만큼 많은 샘플을 분석하기 어렵다.
3. 2번으로 인해 발생하는 문제로, 통계 자료의 신뢰성이 떨어지게 된다.

3. 새롭게 고안한 방법

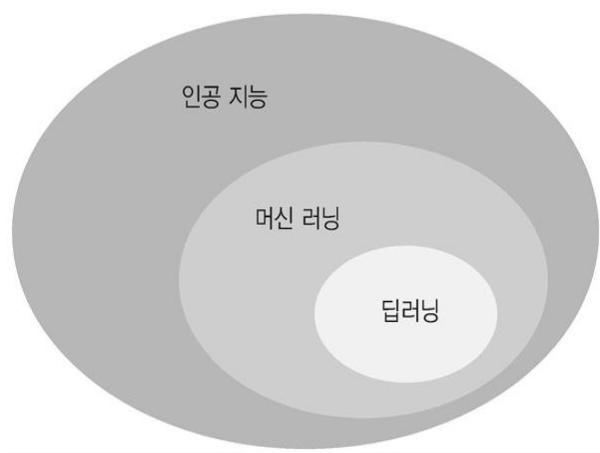
우리는 기존 방법의 문제점을 깨닫고 이것을 개선하기 위한 방법을 고안하였다. 바로 딥러닝과 빅데이터 기술을 사용하는 것이다. 병충해에 걸린 생물 사진을 어떤 병에 걸렸는지 분류하는 딥러닝 모델을 학습시키는 것이다.

<기존 조사 방법의 문제점의 해결>

1. 고급인력인 전문가는 초기 학습시 레이블링에만 필요하다. 단지 사진만 있으면 된다.
2. 사람이 수 시간을 들여야 할 수 있는 수만장의 이미지 분류를 단 1~2초 안에 수행할 수 있다.
3. 수만장의 이미지를 수 초 안에 정보화된 형태로 분류해 쉽게 신뢰성 있는 통계 자료를 만들 수 있다.

이론적 배경

1. 인공 지능



‘인공 지능이란 인간의 학습능력, 추론능력, 지각 능력, 자연언어의 이해능력 등을 컴퓨터 프로그램으로 실현한 기술이다’ -wikipedia. 우리는 4차 산업혁명 시대의 일상에서 딥러닝, 머신러닝, 인공지능이라는 말을 많이 쓰지만 정확한 경계를 알지 못하는 경우가 많다. 좌측 사진에서 세 단어의 관계를 볼 수 있다.

먼저 인공지능은 매우 범위가 넓은 단어이다. 인간이 가지는 능력을 모방하면 모두 인공지능이라고 말할 수 있다. 머신러닝은 인공 지능 분야 중에서 ‘스스로’ 자신의 기능을 빅데이터 등을 이용해 향상하는 기술을 말한다. 마지막으로 딥러닝은 머신 러닝 분야 중에서 인간의 뇌를 모방해 만든 기술인 ‘인

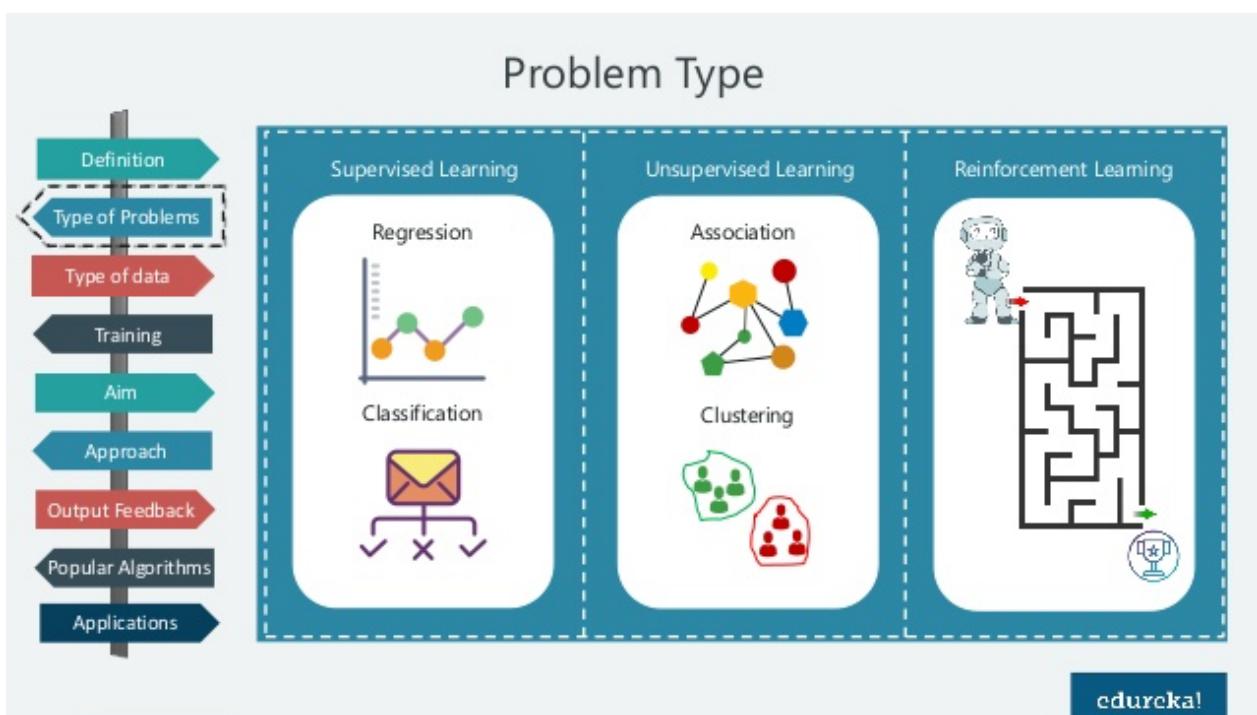
‘공 신경망’을 사용하는 기술을 말한다. 인공 신경망의 특성 때문에 딥 러닝은 비선형적인 데이터에서 대다수의 머신 러닝 기술들보다 더 잘 작동하는 경향을 보인다.

2. 머신러닝(딥러닝)의 3가지 분류

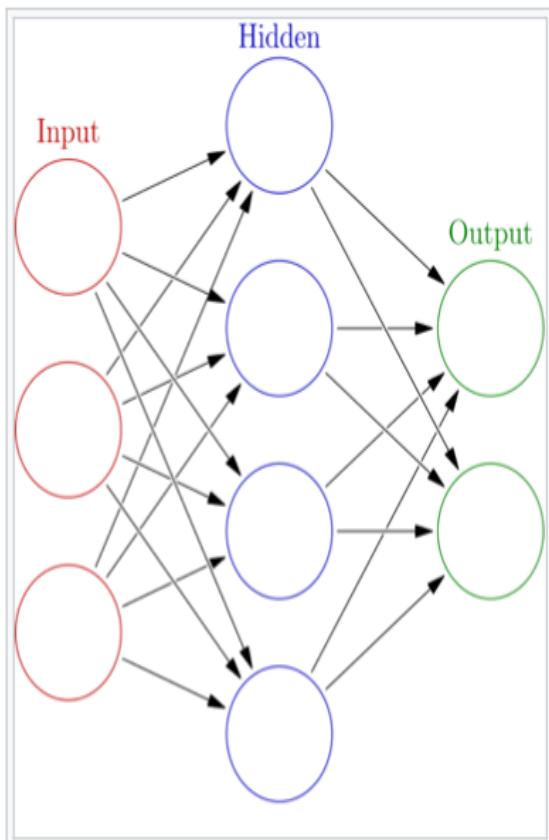
첫 번째 분류는 가장 일반적이고 직관적인 형태인 지도학습이다. 입력-정답 쌍으로 이루어진 데이터에 대해 학습하고 학습에 사용되지 않는, 혹은 정답을 모르는 입력에 대해 정확한 값을 예측하는 것이다. 분류와 회귀는 지도학습에 속한다.

두 번째 분류는 비지도학습이다. 정답이 없는 데이터 속에서 패턴이나 구조를 스스로 찾아낸다. 군집화나 생성 모델이 이 분류에 속한다.

마지막 분류는 강화학습이다. 강화학습에는 환경과 에이전트가 있고, 에이전트는 환경으로부터 상태를 입력받아 행동을 하고 보상을 받는다. 이 보상을 최대화시키는 방향으로 에이전트를 학습시키는것이 목적이다. 알파고/알파스타와 같은 게임인공지능이 이에 속한다.



2. 딥러닝과 인공신경망



‘인공 신경망이란 기계학습과 인지과학에서 생물학의 신경망에서 영감을 얻은 통계학적 학습 알고리즘이다’ -wikipedia. 인공 신경망은 실제 인간의 신경망이 뉴런의 연결으로 이루어지듯이, 노드들이 연결되어 신경망을 이룬다. 입력 노드들이 은닉 노드들에 연결되고, 출력 노드로 나온다.

각 노드에서는 $h(wx + b)$ 연산이 일어난다. $h(x)$ 는 입력과 출력이 비선형적인 관계를 가질 때에도 올바른 관계를 해석할 수 있게 하는 “활성화 함수”이다. 항상 비선형적인 함수를 사용해야 한다. $wx + b$ 는 한 노드에서 일어나는 연산이다.

실제 인공 신경망을 구현할 때에는 다음과 같이 행렬곱으로 나타내어진다.

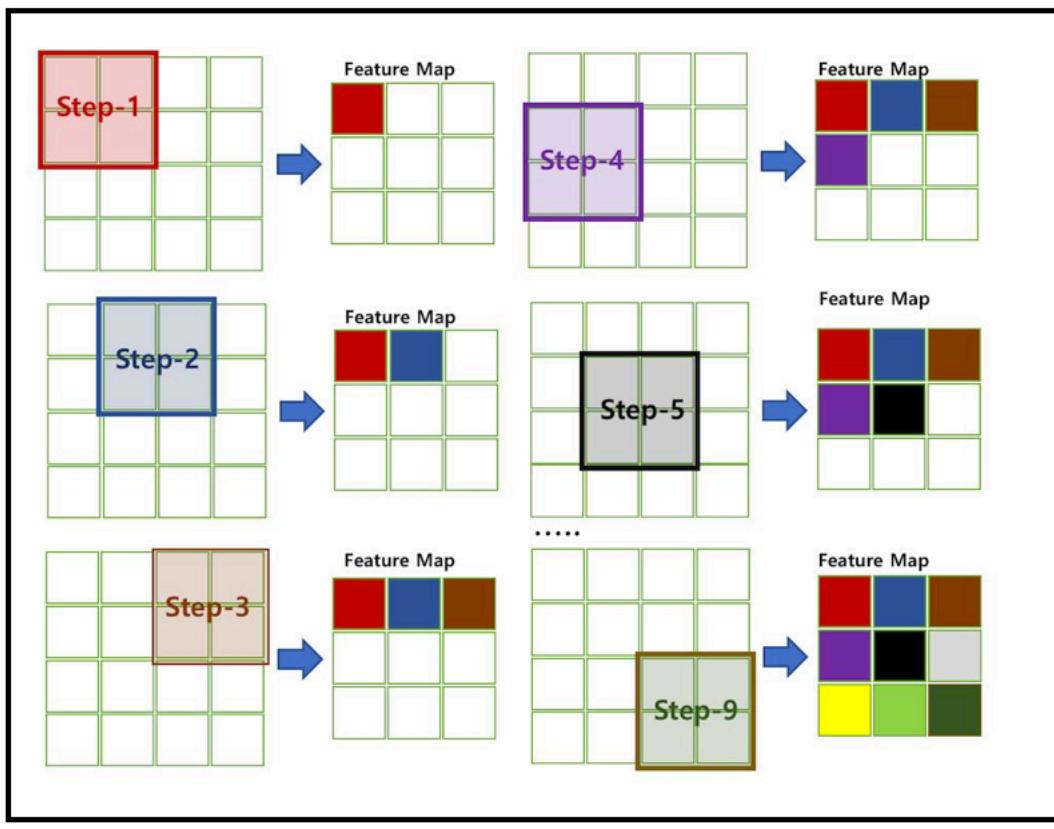
Input Data	Neuron Weights	Outputs Equations
$[X_0 \ X_1 \ \dots \ X_N]$	$* \begin{bmatrix} A_0 & B_0 & C_0 \\ A_1 & B_1 & C_1 \\ \dots & \dots & \dots \\ A_N & B_N & C_N \end{bmatrix}$	$\begin{bmatrix} Y_A = X_0A_0 + X_1A_1 + X_2A_2 \\ Y_B = X_0B_0 + X_1B_1 + X_2B_2 \\ Y_C = X_0C_0 + X_1C_1 + X_2C_2 \end{bmatrix}$

행렬이 공간의 변환이라는 관점에서 보면 딥 러닝이 하는 연산은, 입력 공간을 출력 공간으로 비선형적인 변환을 하는 작업이다. 하나의 노드로 표현되는 행렬의 원소가 너무 많다 보니 인공 신경망이 어렵게 입력과 출력 사이의 관계를 해석하는지 쉽게 알 수 없다(Black-Box)라는 문제점도 있다.

3. 이미지 인식의 최강자-CNN

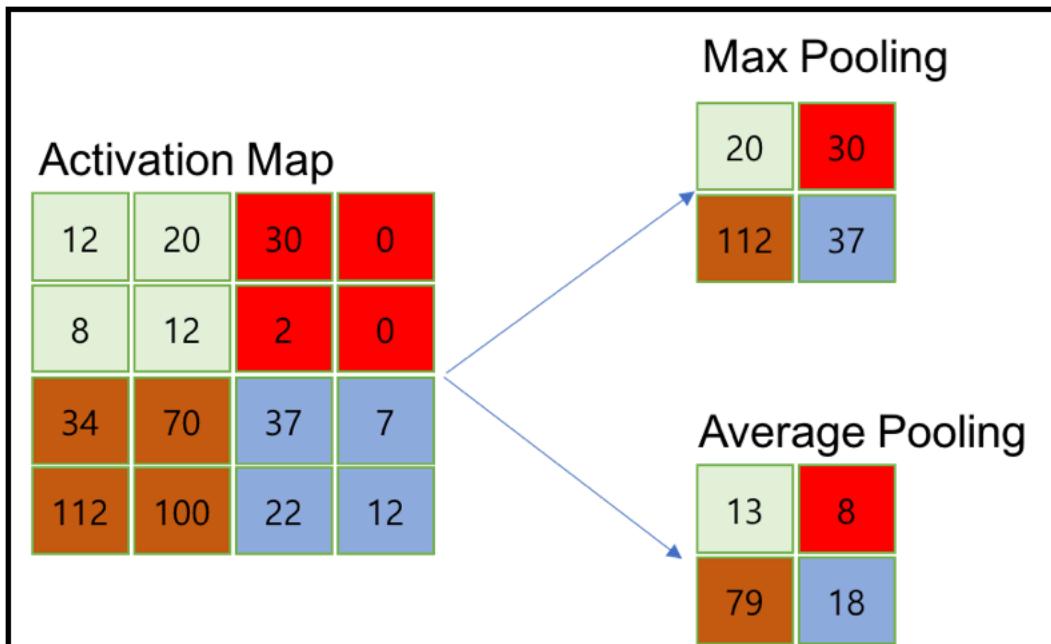
일반 신경망에 비해 이미지 인식에 최적화된 신경망이 있다. 바로 CNN(Convolutional Neural Network, 합성곱 신경망)이다. 얀 르쿤에 의해 1989년 처음 제안되었다. 현재까지 CNN을 기반으로 많은 신경망들이 개발되어 쓰이고 있다. 아래에서는 CNN의 작동 방식을 설명한다.

1. 합성곱 필터를 이용해 이미지의 특징을 추출한다. 필터가 이미지의 부분들을 일정한 간격으로 순회하며 이미지 행렬에 곱해진다. 그것을 모아서 특징을 추출한다.



필터를 사용하는 것은 학습하는 가중치의 수를 줄이는 장점이 있다. 상단 그림의 예제를 보자. 만약 우리가 필터 2×2 필터 4개를 사용한다면 학습해야 하는 노드의 수는 $2 \times 2 \times 4 = 16$ 개이다. 하지만 일반 신경망을 사용한다면 $16 \times 9 = 144$ 개의 노드를 학습해야 한다. 만약 이미지의 크기가 4×4 가 아닌 실제 상황에서 사용되는 1920×1080 정도의 크기라면 이 차이는 엄청날 것이다.

2. 풀링 계층을 통해 차원을 축소한다.



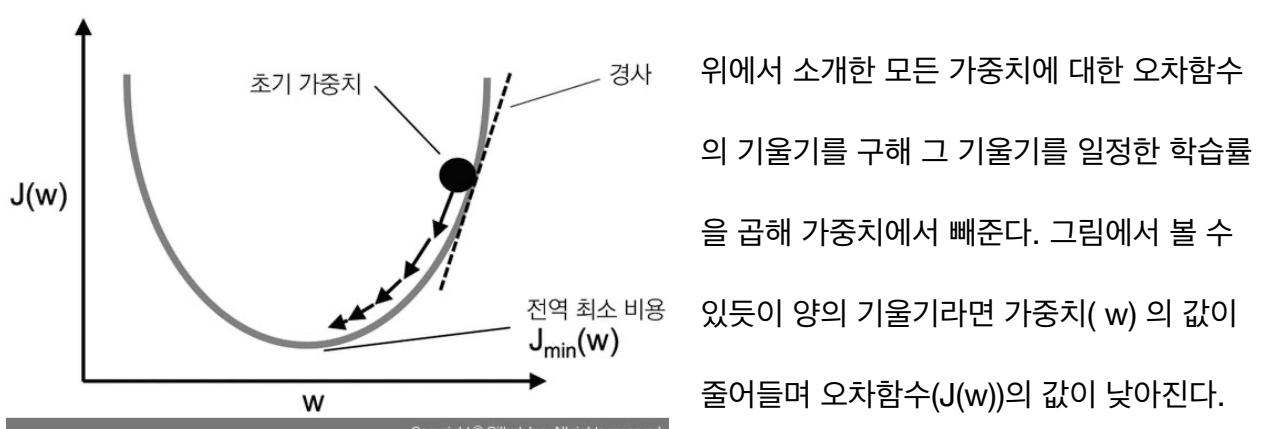
역시 학습해야 할 노드의 수를 줄이기 위해, 위의 합성곱 층을 통과한 특징 계층은 풀링 계층에 의해 차원 축소될 필요가 있다. 풀링 계층은 학습하는 파라미터가 없다. 그저 특정 사이즈의 필터를 돌리며 그 필터 중 가장 큰 값을 뽑거나 평균값을 뽑는 연산을 한다. 이렇게 합성곱 층과 풀링 계층을 몇 번 반복한 후 축소된 이미지의 특징 행렬은 모든 노드가 연결된 일반 신경망에 들어가고 출력 계층으로 연결된다.

CNN이 이미지를 해석하는 방법은 인간이 이미지를 해석하는 방법과 닮아 있다. 인간도 얼굴과 같은 특징점이 존재하는 이미지를 볼 때, 전체를 보기보다는 특징점을 본다. 예를 들어 얼굴의 눈, 코, 입과 같은 특징점을 주의깊게 살펴보게 된다.

4. 딥러닝이 배우는 방법, 경사 하강법

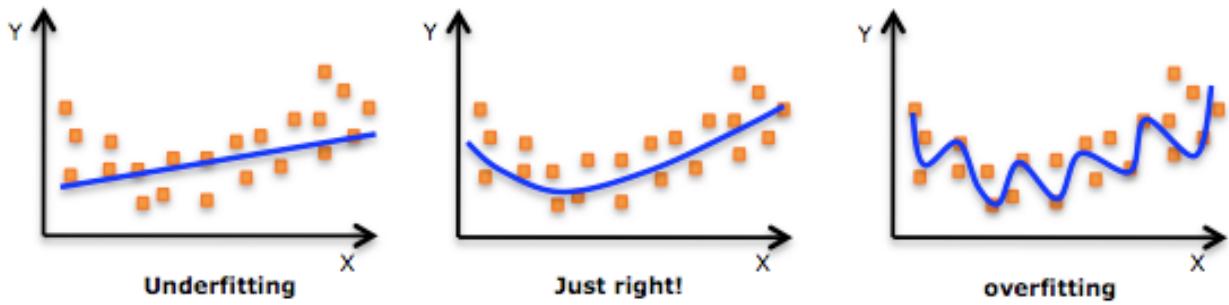
우리가 인공 신경망을 만드는 목적은 정답을 모르는 입력 데이터에 대하여 정답을 예측하기 위함이다. 그 작업을 수행하기 위해서는 인공 지능에게 먼저 입력-정답으로 이루어진 데이터를 학습시켜 주어야 한다. 이 과정을 가능하게 하는 것이 바로 ‘경사 하강법’이다.

입력 데이터 x 가 있다고 하자. 이것을 인공 신경망에 넣으면 y_p 라는 예측값을 뽑아낸다. 예측값인 y_p 와 실제값인 y 의 차이가 줄어드는 쪽으로 신경망을 학습하면 나중에 학습 데이터에 없던 입력값에 대해서 정확한 예측값을 뽑아낼 수 있을 것이다. 대표적인 오차함수로는 MSE (평균제곱오차)가 있다. 경사 하강법은 이런 오차함수를 이용해 만들어낸 오차값을 줄이는 방법이다.



5. 과적합

인공 신경망의 한계점 중 하나는 과적합이다. 학습에 사용된 데이터에 너무 심하게 적합되어, 실제로 예측해야 할 주어지지 않은 데이터에 대해서는 성능이 잘 나오지 않는 것이다. 이런 과적합을 방지하기 위해 신경망 노드를 확률적으로 사용 중지하며 학습하는 Dropout 방법이나, 데이터셋 중 학습에 사용될 데이터와 정답을 알지만 학습에 사용하지 않는 데이터를 분류해 검증에 사용하지 않는 방법이 주로 사용된다.



인공 신경망을 사용한 식물 잎 병충해 분류기

1. 구현에 사용한 컴퓨터 언어와 라이브러리

- 프로그래밍 언어: Python 3.7
- 인공 신경망 사용을 위한 TensorFlow
- 분산 컴퓨팅 처리를 위한 Ray
- Csv 데이터를 다루기 위한 pandas
- 행렬/수학 연산을 위한 numpy
- 시각화를 위한 matplotlib
- 이미지 로딩을 위한 PIL
- 이외 파이썬 기본 모듈

<소스 코드>

```

import os
import ray
import random
import numpy as np
import pandas as pd
from PIL import Image
from tensorflow.keras import *
import matplotlib.pyplot as plt
from mpl_toolkits.axes_grid1 import ImageGrid
from keras_radam.training import RAdamOptimizer

```

2. 데이터셋

데이터프레임 확인하기																																									
image_id: 각 이미지의 고유 id. 레이블과 이미지를 매칭하는 데 쓴다																																									
healthy: 식물이 건강한 경우.																																									
multiple_diseases: 식물이 복합적인 병을 가진 경우.																																									
rust: 식물이 녹병균에 걸린 경우.																																									
scab: 식물 잎에 딱지가 많이 앓은 경우																																									
<code>train_df.head()</code>																																									
<table border="1"> <thead> <tr> <th></th> <th>image_id</th> <th>healthy</th> <th>multiple_diseases</th> <th>rust</th> <th>scab</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Train_0</td> <td>0</td> <td></td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>Train_1</td> <td>0</td> <td></td> <td>1</td> <td>0</td> </tr> <tr> <td>2</td> <td>Train_2</td> <td>1</td> <td></td> <td>0</td> <td>0</td> </tr> <tr> <td>3</td> <td>Train_3</td> <td>0</td> <td></td> <td>0</td> <td>1</td> </tr> <tr> <td>4</td> <td>Train_4</td> <td>1</td> <td></td> <td>0</td> <td>0</td> </tr> </tbody> </table>							image_id	healthy	multiple_diseases	rust	scab	0	Train_0	0		0	0	1	Train_1	0		1	0	2	Train_2	1		0	0	3	Train_3	0		0	1	4	Train_4	1		0	0
	image_id	healthy	multiple_diseases	rust	scab																																				
0	Train_0	0		0	0																																				
1	Train_1	0		1	0																																				
2	Train_2	1		0	0																																				
3	Train_3	0		0	1																																				
4	Train_4	1		0	0																																				

입력 데이터 - 1821개의 식물 잎 이미지

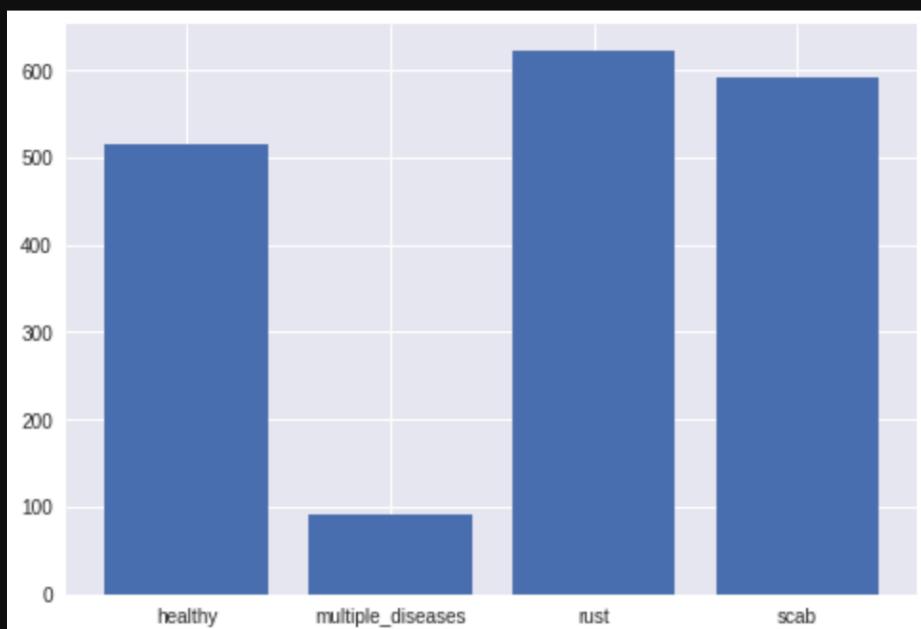
출력 데이터 - 해당하는 입력 잎의 상태 (건강
함, 복합적인 병, 녹병, 딱지 앓은 잎)

좌측은 정답 레이블이 담긴 csv 파일을
pandas를 이용해 pandas의 자체 자료구조
인 DataFrame으로 로딩한 모습이다.
4개의 잎 상태와 image id가 있는 모습을 볼
수 있다.

데이터셋의 분포를 확인해보니, 다른 label에 비해 multiple diseases에 포함되는 이미지가 적다는
것을 확인할 수 있었다.

<소스 코드>

```
plt.bar(train_df.columns[1:], [train_df[x].sum() for x in train_df.columns[1:]])  
<BarContainer object of 4 artists>
```



이는 학습을 불안정하게 만들 수 있어, 데이터셋의 불균형(imbalance)를 해결하는 가장 간단한 방법

인 해당 레이블의 이미지를 복사하여 여러 번 붙여넣는 방식으로 불균형을 해결해 주었다.

<소스 코드>

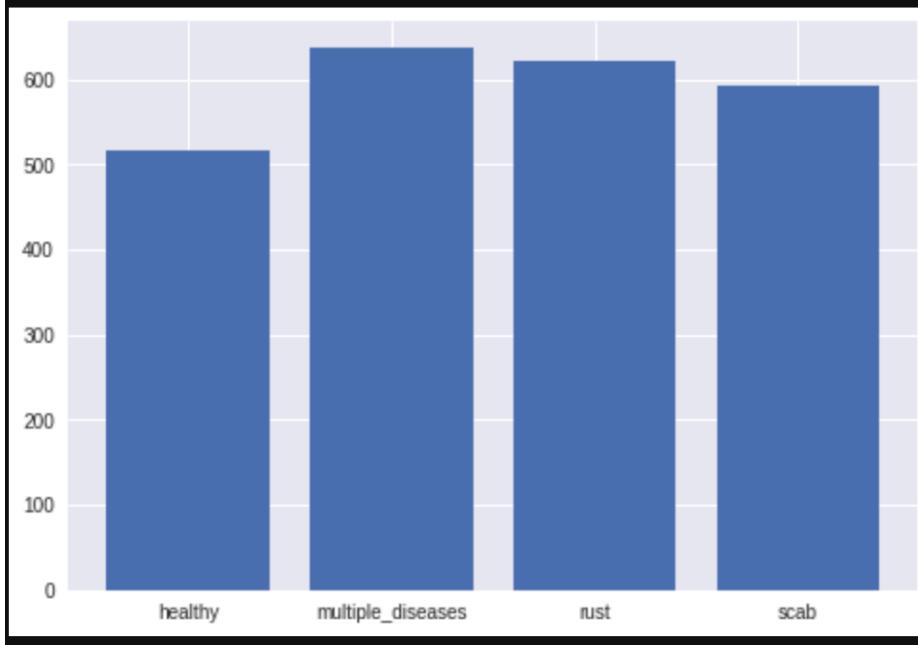
```
n = np.mean([[len(list(train_df[train_df[x] == 1].index))
    for x in ["healthy", "rust", "scab"]]]) //
    len(list(train_df[train_df["multiple_diseases"] == 1].index))

appender = pd.DataFrame(train_df[train_df["multiple_diseases"] == 1])

for _ in range(int(n)):
    train_df = train_df.append(appender)

train_ids = train_df["image_id"].to_list()

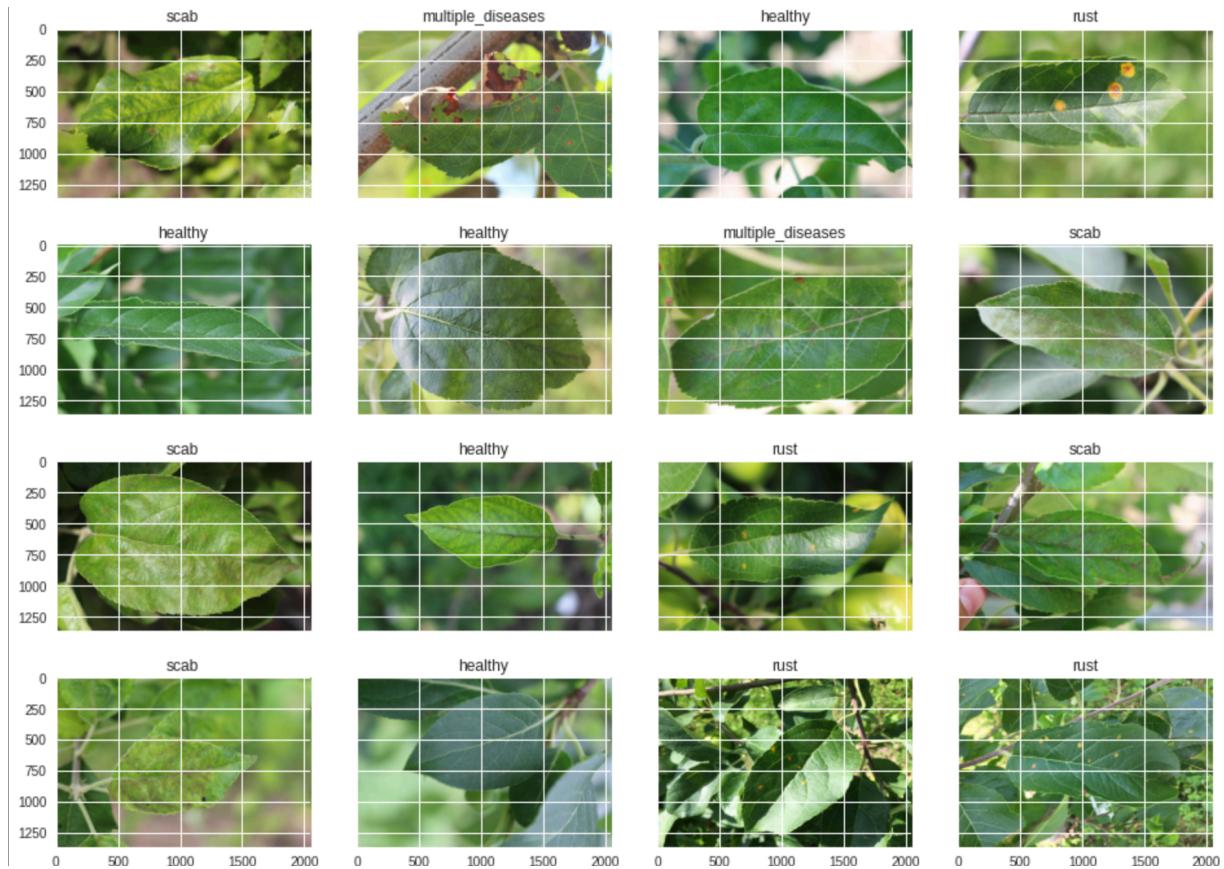
plt.bar(train_df.columns[1:], [train_df[x].sum() for x in train_df.columns[1:]])
train_df = train_df.drop_duplicates()
```



데이터셋의 분포가 안정된 모습을 볼 수 있다. 첫 실험에 데이터 Imbalance를 잡아주지 않고 학습했

더니 매우 불안정해서 도입하게 되었다.

다음은 이미지를 확인해 본다.

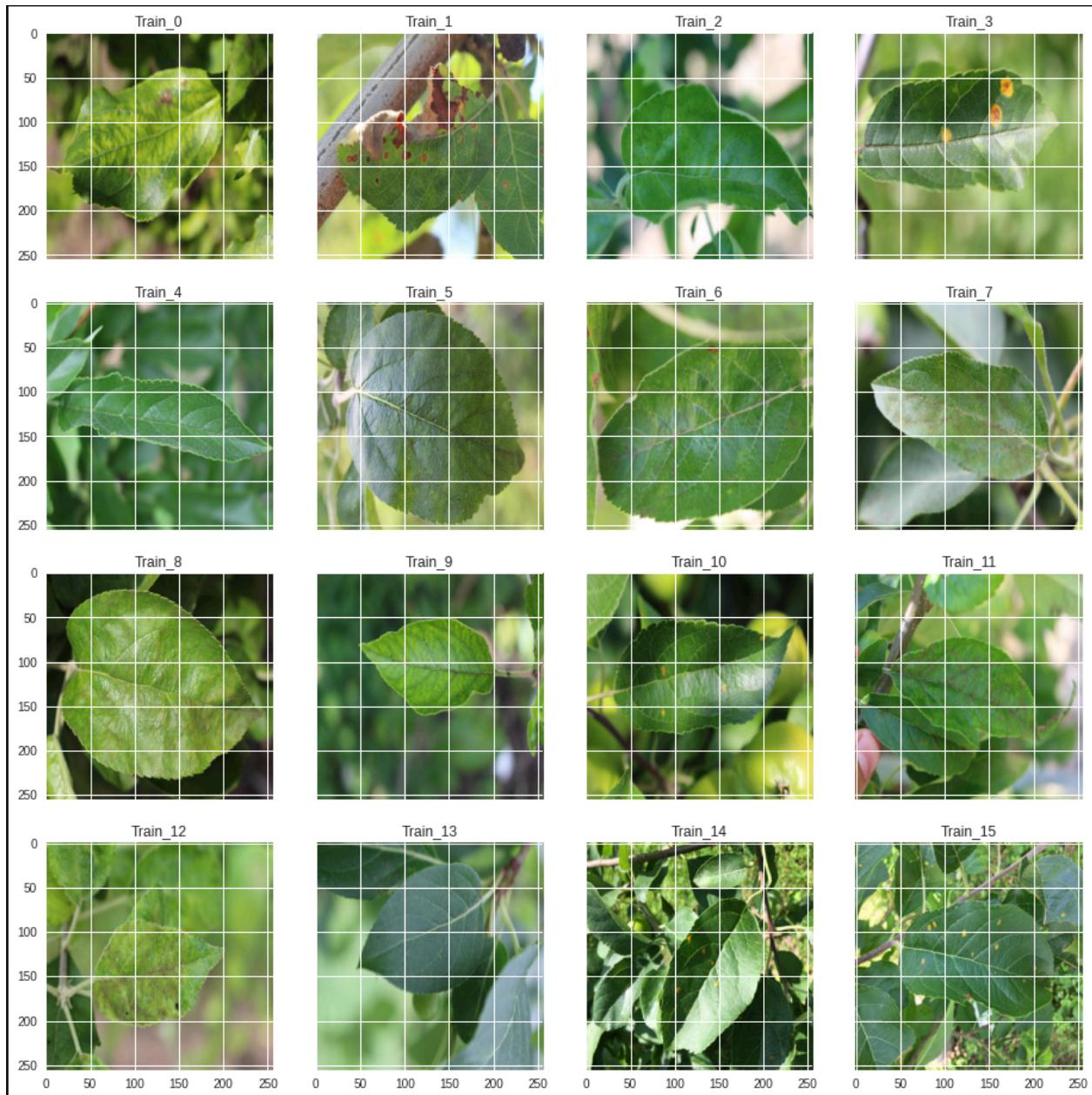


이미지가 고해상도인 것을 볼 수 있다. 하지만 고해상도의 이미지는 인공 신경망 학습 시 너무 많은 컴퓨팅 자원을 사용하여 비효율적이다. 또한 이미지의 픽셀의 값들이 0~255 사이의 값으로 나타내져 있지만 인공 신경망의 노드의 가중치와 편향의 값들은 0~1 사이이다. 이것은 경사하강법을 사용할 때 경사가 발산해버리는 결과를 초래할 수 있다. 따라서 이미지의 사이즈를 256x256 으로 줄이고, 픽셀의 값을 255를 나눠 0~1 사이의 값으로 만들어준다.

<소스 코드>

```
@ray.remote
def load_preprocess_image(image_id, resize_to=IMAGE_SIZE):
    path = os.path.join(image_dir, image_id + ".jpg")
    image_arr = np.array(Image.open(path).resize(resize_to)) / 255
    return image_arr

def load_label(image_id):
    return train_df.loc[train_df["image_id"] == image_id][train_df.columns[1:]].values.squeeze()
```

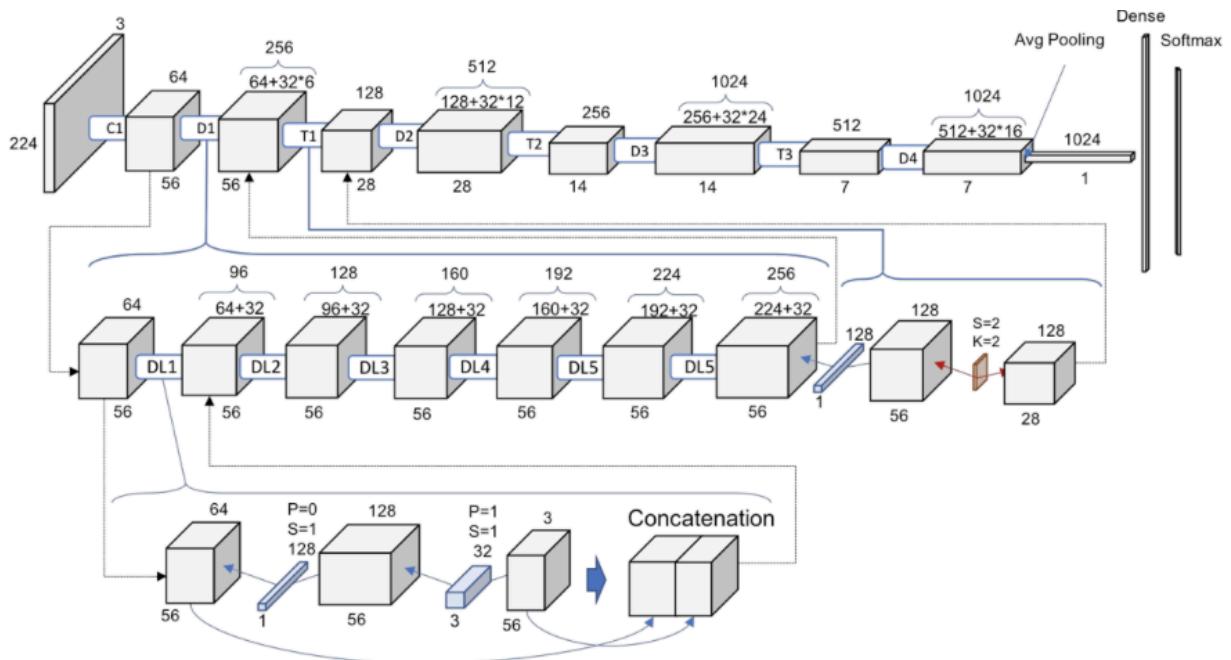


이미지가 전처리 된 것을 확인할 수 있다. 이것이 최종적으로 인공 신경망이 보는 형태이다.

3. 인공 신경망 모델

위에서 소개한 CNN(합성곱 신경망)을 기반으로 한 DenseNet 모델을 사용한다. Hwang에 의해 2016년 제안되었다. 학습 시간을 단축하기 위해 ImageNet 데이터셋에 대해 사전학습 된 모델을 우리가 가진 데이터에 대해 전이학습하였다. ImageNet이라는 매우 큰 이미지 데이터에 대해 사전학습 해 모델이 ‘이미지에 대한 이해’를 갖게 된 상태에서 우리 데이터셋을 이용하면 좀 더 효과적인 학습이 가능하다.

아래는 DenseNet 모델의 구조이다.



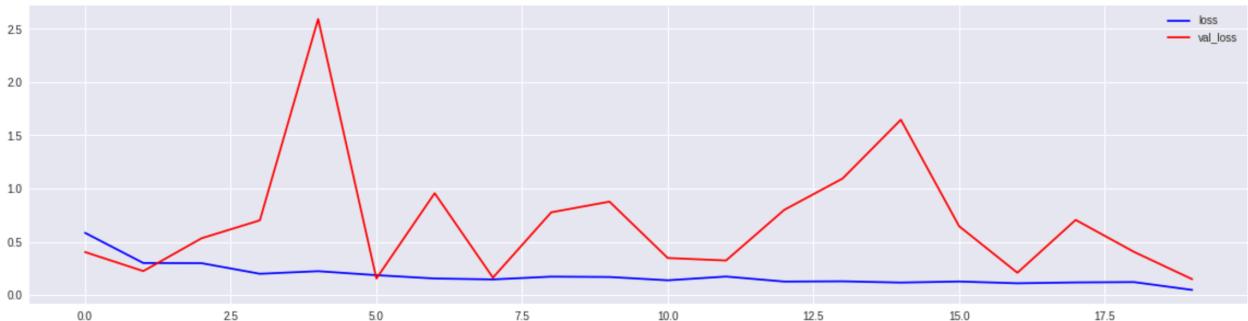
4. 학습 결과

위에서 소개한 경사 하강법을 학습률을 자유자재로 조절하는 형태로 개선한 RAdam이라는 경사 하강법을 사용하여 학습하였다. 학습이 불안정하지만 빠르고 정확하기 때문에 컴퓨팅 자원이 부족한 우리의 실험 환경에 적합하다고 판단했다. 학습 결과를 보면 비교적 적은 데이터에도 불구하고 높은 정확도를 보여주는 것을 알 수 있다. 전이학습의 효과라고 볼 수 있다. 수십만 장의 ImageNet Data를 통해 먼저 이미지 데이터에 대한 이해를 신경망에게 준 후, 우리의 식물 병충해 데이터를 학습해주는 것이다.

<오차함수 값 그래프>

-val_loss: 밸리데이션 데이터에서의 오차 함수 값

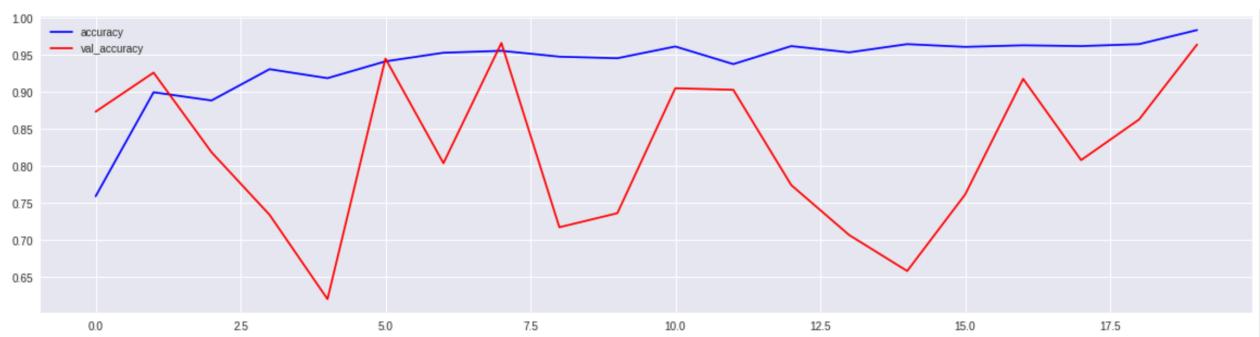
-train_loss: 학습 데이터에서의 오차 함수 값



<정확도 값 그래프>

-val_loss: 밸리데이션 데이터에서의 정확도 값

-train_loss: 훈련 데이터에서의 정확도 값



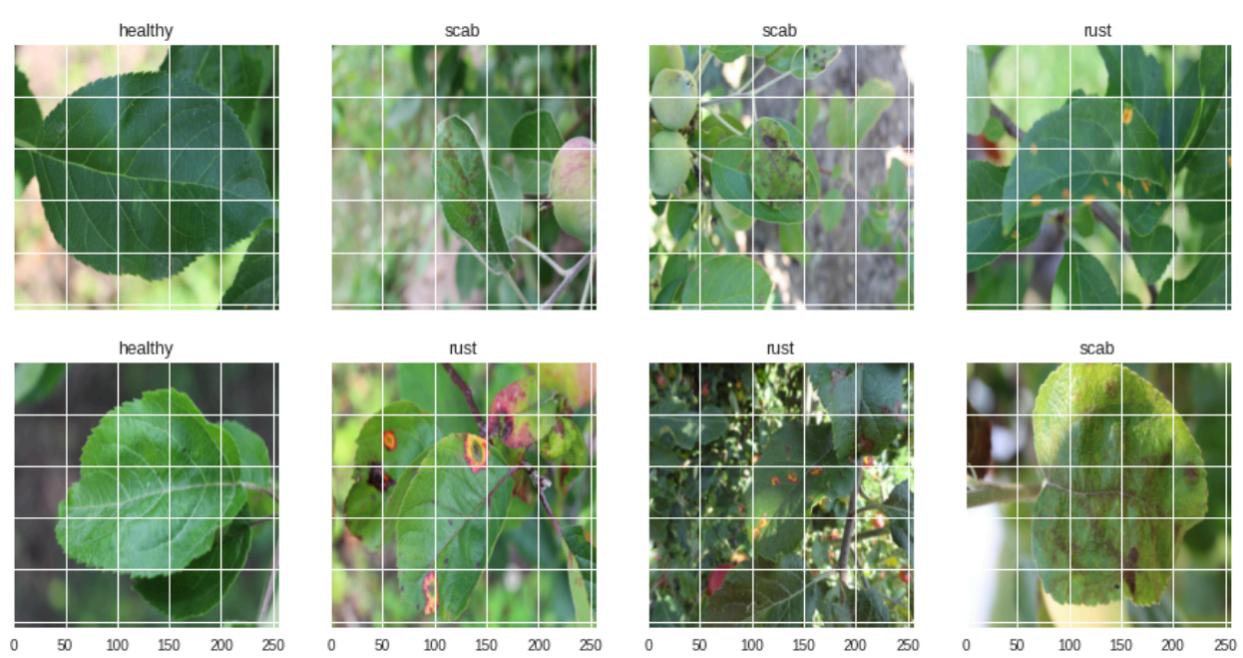
첫 에폭(데이터셋 순환) 만에 밸리데이션 데이터 (학습에 사용되지 않고, 학습을 검증하기 위한 데이터)

에서 87%의 정확도를 달성한 것을 볼 수 있다. 그렇게 오르락내리락하다 결국 밸리데이션 데이터에

서 96%의 정확도를 달성했다. 데이터 제공처인 [kaggle](#)이 제공한 추가 테스트 데이터에 대해서는

92.7%의 정확도를 달성했다.

결론



사용한 데이터셋과 인공 신경망 모델을 이용해 성능이 괜찮은 식물 병충해 분류기를 만들 수 있었다.

위의 사진에서 볼 수 있듯이 학습에 사용되지 않은 데이터셋에 대해 정확한 예측 결과를 낼 수 있었다. 이러한 모델이 산업현장에 적용되면, 기존 전문가가 하던 일을 더욱 효율적이고 경제적으로 정확하게 할 수 있을 것이다. 또한 그런 전문가를 고용하기 힘든 개발도상국이나 환경 관련 벤처기업 등에서도 이런 인공지능과 컴퓨터만 있으면 쉽게 자연환경과 생물다양성의 실태를 조사하고 보전하는 데에 사용할 수 있을 것으로 예상된다.

딥러닝을 이용한 이미지 예측은 식물 뿐만 아니라 동물, 혹은 생태계의 사진에도 유연하게 적용할 수 있는 방법론이다. 앞으로 딥러닝이 생물다양성에 기여할 부분은 무궁무진할 것이다. 이것은 딥러닝과 같은 최첨단기술이 어떻게 인간과 자연환경에 도움이 되는지 보여주는 좋은 예다. 이 프로젝트는 우리에게 딥러닝 기술을 우리 생활과 지구, 환경을 개선하는 데에 사용할 동기부여가 되었다. 이렇게 딥러닝과 같은 앞으로 생물다양성에서의 딥러닝/인공지능의 활약을 기대하는 바이다.

추가로 연구할 점

- 데이터셋을 구하게 된다면 동물 등 다른 생물군의 병이나 생태계 데이터에 대해 추가로 연구를 진행하는 것이 의미가 있을 것 같다.
- 사진을 모으는 방법까지 무인 항공기(UAVs) 등을 사용하여 많은 사진을 수집한 다음, 딥러닝을 이용해 세그멘테이션을 수행해 자동화하는 방법을 사용한다면, 조금 더 실용적인 기술을 만들 수 있을 것이다.
- 좀 더 나은 컴퓨팅 자원이 마련된다면, 같은 데이터셋에 대해 더 큰 인공 신경망 모델을 사용하여 결과값을 확인해 보면 나은 결과가 나올 수 있을 것 같다.
- 데이터셋의 불균형을 제거하는 더 나은 방법에 대해 알아보고 적용하면 현재 결과보다 진보된 결과를 기대할 수 있겠다.
- 실험에 사용된 RAdam 외에 다른 경사하강법의 변형에 대해서도 실험해보면 풍부한 결과를 기대할 수 있다.
- 사전학습이 되지 않은 신경망에 대해서 이 데이터셋으로만 학습시킨 다음 현재 학습결과와 비교/대조하면 사전학습의 효과를 더 부각할 수 있을 것이다
- 현재 학습에서 사용된 지도 학습(Supervised Learning) 이외에도 강화 학습(Reinforcement Learning) 비지도 학습(Unsupervised Learning)을 사용하여 데이터셋을 다르게 해석하는 방법에 대해 연구하면 새로운 결과를 기대할 수 있다.
- 이런 모델을 Python 환경에서만 사용하는 것을 넘어 안드로이드나 IOS 혹은 웹앱으로 배포하면 실제 산업현장에 직접적인 도움을 더 많이 줄 수 있을 것이다.

소스 코드

보고서에 사용된 소스 코드는 팀에서 직접 작성하였으며, [Github](#)에서 찾으실 수 있습니다.
(<https://github.com/Yeachan-Heo/G.G.G-at-GNBS>)

참고 문헌

[케라스 창시자에게 배우는 딥러닝\(2018\), 프랑소와 솔레 저, 길벗 출판](#)

[밀바닥부터 시작하는 딥러닝 1\(2017\), 사이토 고키 저, 한빛미디어 출판](#)

[Alex Krizhevsky\(2012\), ImageNet Classification with Deep Convolutional Neural Networks, NeurIPS 2012](#)

[Diederik P. Kingma\(2015\), ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION, ICLR 2015](#)

[Gao Huang\(2016\), Densely Connected Convolutional Networks, CVPR 2017](#)

[Liyuan Liu\(2020\), On the Variance Of Learning Rate and Beyond](#)

[Ranjiita Thapa\(2020\), The Plant Pathology Challenge 2020 data set to classify foliar disease of apples, Applications in Plant Sciences](#)

[wikipedia\(deep learning\)](#)

[wikipedia\(AI\)](#)

[wikipedia\(ANN\)](#)

[세계일보\(2020-10-17\) 기후변화 위기 대응, 생물다양성 보전 '최전선'](#)

[환경미디어\(2020-02-10\) AI, 원인과 대안으로 떠오른 환경문제 해결사?](#)

[사이언스모니터\(2019-02-26\) 스마트팜 활용 국내 자생식물 보존 및 지속가능 자원화](#)

[환경미디어\(2017-10-31\) 인공지능 시대에 걸맞은 똑똑한 환경기술: “4차 산업혁명 기술에 의해 환경정책도 바뀔 것”](#)

[환경미디어\(2020-06-01\) 생물다양성 손실이 펜데믹 더욱 증가시켜](#)

[한국일보\(2020-01-28\) 생물다양성과 질병](#)

[한겨례\(2020-10-04\) 중남미-카리브해 척추동물 94%가 이미 멸종](#)

[오마이뉴스\(2020-10-14\) 기후위기를 생물다양성으로 뚫고 가자](#)

[THE DAILYPOST\(2020-09-15\) 반세기 만에 지구 야생동물의 68%가 사라졌다.](#)

[Science Codex\(2020-10-20\) Artificial Intelligence reveals hundreds of millions of trees in the Sahara](#)

[EurekAlert\(2020-09-28\) Artificial intelligence can help protect orchids and other species](#)

[EuroNews\(2019-12-24\) AI FROM GOOGLE IS HELPING IDENTIFY ANIMALS DEEP IN THE RAINFOREST](#)