



License Plate Recognition

차량 번호판 인식 프로그램
17681005 김규영
17681026 장정우



Table of Contents

01

데이터 분석 주제

- 팀원 역할
- 번호판 인식
- 아이디어

03

신경망 모델

- 신경망
- 학습 과정
- 손실도와 정확도

02

데이터 시각화

- 번호판 인식 시각화

04

마무리

- 시연 동영상
- 팀원별 소감

01

데이터 분석 주제



팀원 역할



김규영

17681005

자료 수집

데이터 분석

손실도 및 정확도 코드 분석

XX DRIVER LICENSE XX



장정우

17681026

아이디어 제안

신경망 및 학습과정 분석

프레젠테이션 제작

XX DRIVER LICENSE XX

아이디어

차량 번호판 인식

딥러닝을 통해 차량
번호판을 학습



학습 후 인식

번호판을 학습 시킨 후
인식



다양한 활용

차량 주차관리 시설,
출입통제가 필요한 장소,
도로교통 방법 카메라를
통한 단속

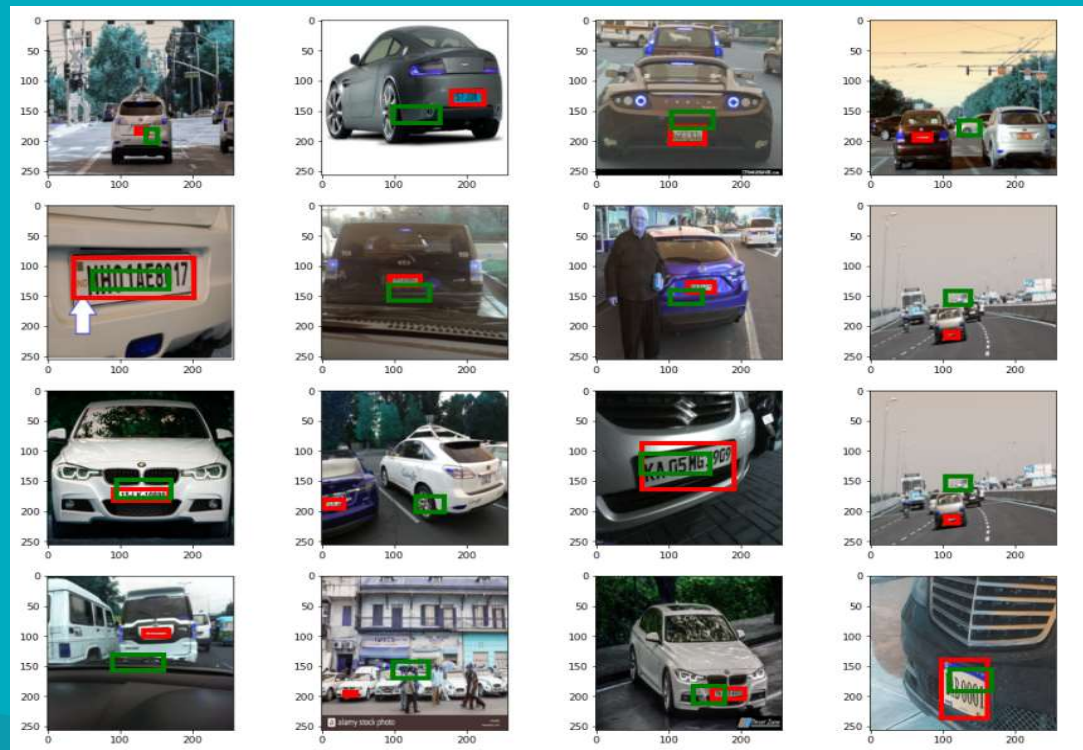


02

데이터 시각화

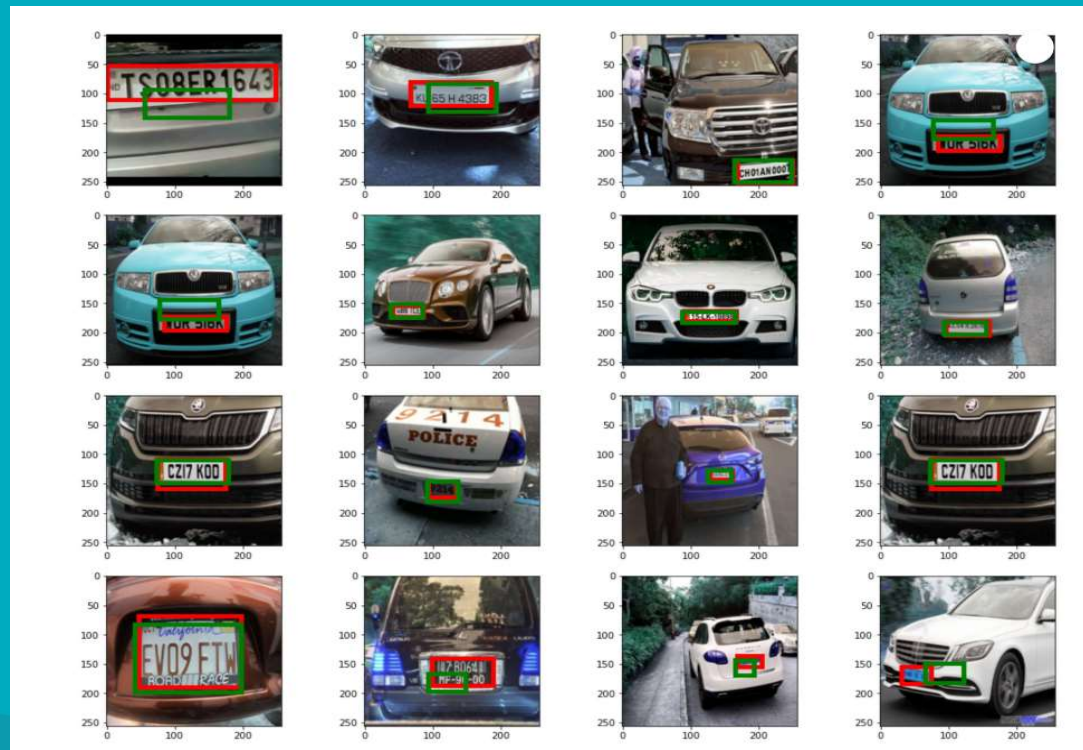


번호판 인식 시각화



*10번의 학습

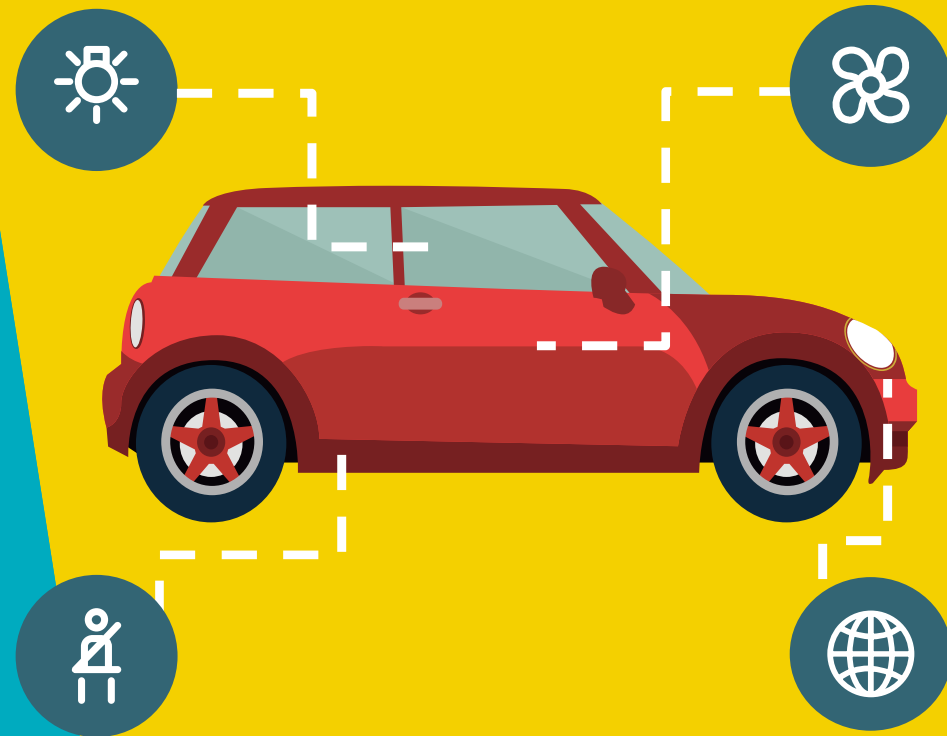
번호판 인식 시각화



*30번의 학습

03

신경망 모델



신경망

```
In [5]: model = Sequential()
model.add(VGG16(weights="imagenet", include_top=False, input_shape=(256,256, 3)))
model.add(Flatten())
model.add(Dense(128, activation="relu"))
model.add(Dense(128, activation="relu"))
model.add(Dense(64, activation="relu"))
model.add(Dense(4, activation="linear"))

model.layers[-6].trainable = False

model.summary()

## relu 경사함수, linear 선형회귀를 통해 예측모델을 만듦.
```

Layer (type)	Output Shape	Param #
vgg16 (Functional)	(None, 8, 8, 512)	14714688
flatten (Flatten)	(None, 32768)	0
dense (Dense)	(None, 128)	4194432
dense_1 (Dense)	(None, 128)	16512
dense_2 (Dense)	(None, 64)	8256
dense_3 (Dense)	(None, 4)	260

```
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
              loss='mse',
              #MSE는 회귀(regression) 용도의 딥러닝 모델을 위한 손실함수
              metrics=["accuracy"])

initial_epochs = 30

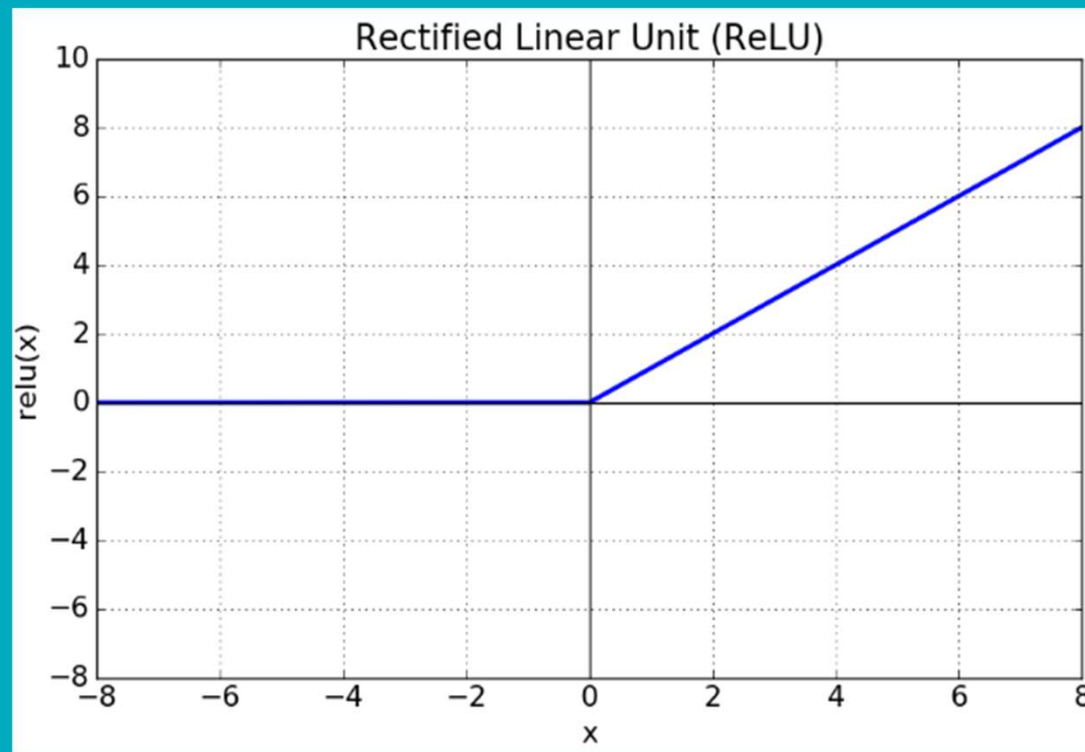
#early_stop = keras.callbacks.EarlyStopping(patience=2)
#과적합 되지않게 early_stop을 정해놨다.

history = model.fit(X_train, y_train_new,
                    validation_data=(X_val, y_val_new),
                    epochs=initial_epochs)
```

*relu 함수
*linear 함수
*mse 함수

신경망

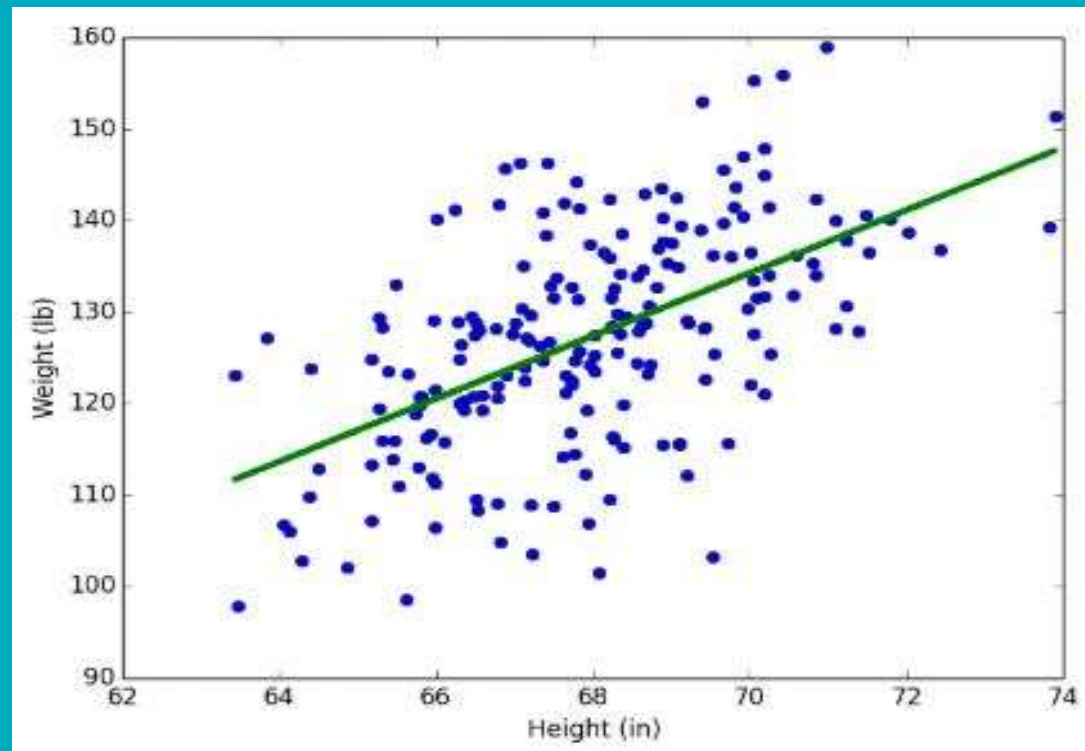
Relu 경사함수



$$f(x) = \max(0, x)$$

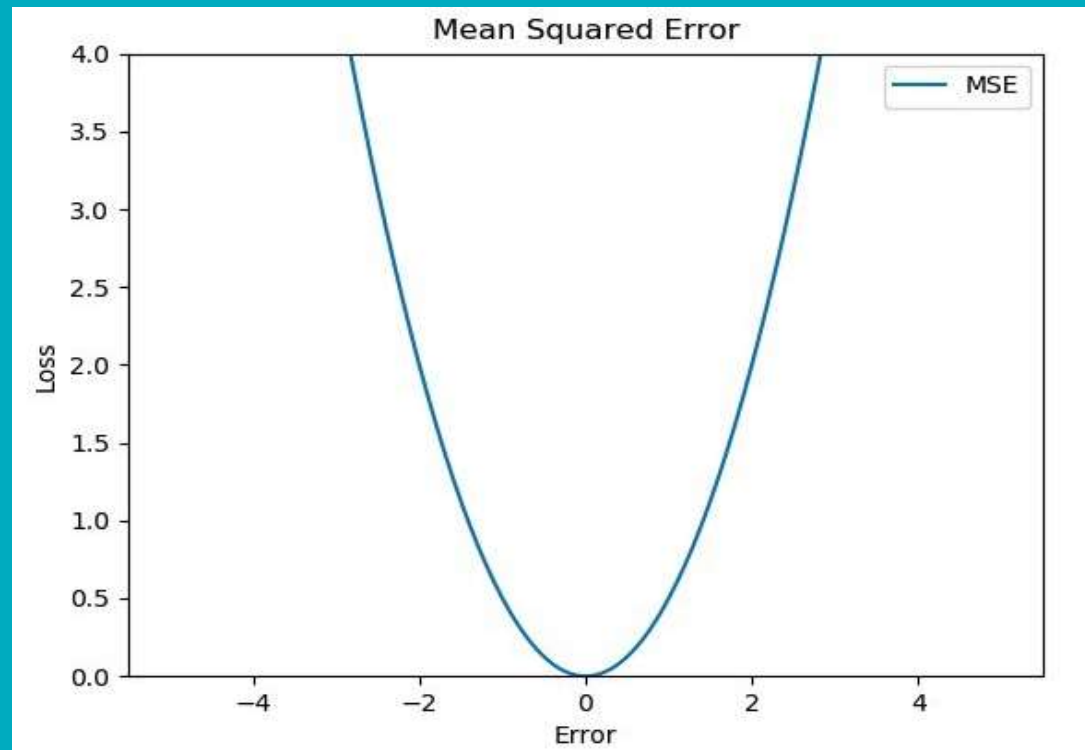
신경망

Linear 선형회귀



신경망

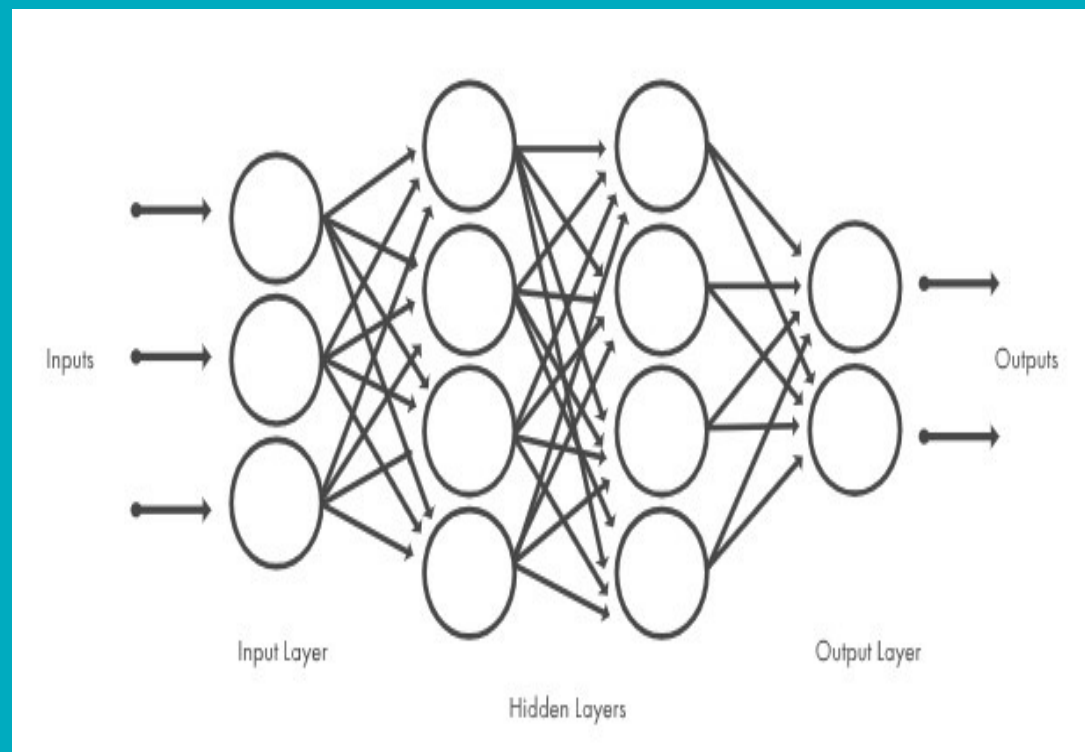
Mse 손실함수



$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (\hat{Y}_i - Y_i)^2$$

신경망

CNN



***Convolutional Neural Network**

학습 과정

Cars108.xml

← → ↻ ⓘ 파일 | C://Users//jungw//Downloads//archiv... 📄 📌 ☆ 🏠 정우

This XML file does not appear to have any style information associated with it. The document tree is shown below.


```
<?xml version="1.0" encoding="UTF-8" ?>
<annotation>
  <folder>images</folder>
  <filename>Cars108.png</filename>
  <size>
    <width>442</width>
    <height>333</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <name>licence</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <occluded>0</occluded>
    <difficult>0</difficult>
    <bndbox>
      <xmin>158</xmin>
      <ymin>216</ymin>
      <xmax>277</xmax>
      <ymax>248</ymax>
    </bndbox>
  </object>
</annotation>
```

```
In [40]: X = []
         y = []
         import matplotlib.pyplot as plt
         for car in coches:
             data_coche = coches[car]
             data_coche_resized = cv2.resize(data_coche, (256,256))
             y_ = data_coche.shape[0]
             x_ = data_coche.shape[1]
             x_scale = (256/x_)
             y_scale = (256/y_)
             box_car = boxes[car]
             xmin = (int(box_car[0]))*x_scale
             ymin = (int(box_car[1]))*y_scale
             xmax= (int(box_car[2]))*x_scale
             ymax= (int(box_car[3]))*y_scale
             img = cv2.resize(data_coche, (256,256))

             plt.imshow(img)
             plt.plot([xmin,xmax], [ymax,ymax], c = "red",linewidth=7.0)
             plt.plot([xmin,xmax], [ymin,ymin], c = "red",linewidth=7.0)
             plt.plot([xmin,xmin], [ymax,ymin], c = "red",linewidth=7.0)
             plt.plot([xmax,xmax], [ymax,ymin], c = "red",linewidth=7.0)
             plt.show()

             X.append(img)
             y.append((xmin,ymin,xmax,ymax))

         ## 각각의 사진의 크기들이 다르기 때문에 학습에 용이하게 맞춰주는 과정
         ## 미리 입력해놓은 정확한 번호판 위치
```



학습 과정

```
In [8]: model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
                    loss='mse',
                    #MSE는 회귀(regression) 용도의 딥러닝 모델을 훈련시킬때 많이 사용되는 손실 함수입니다.
                    metrics=['accuracy'])

initial_epochs = 10

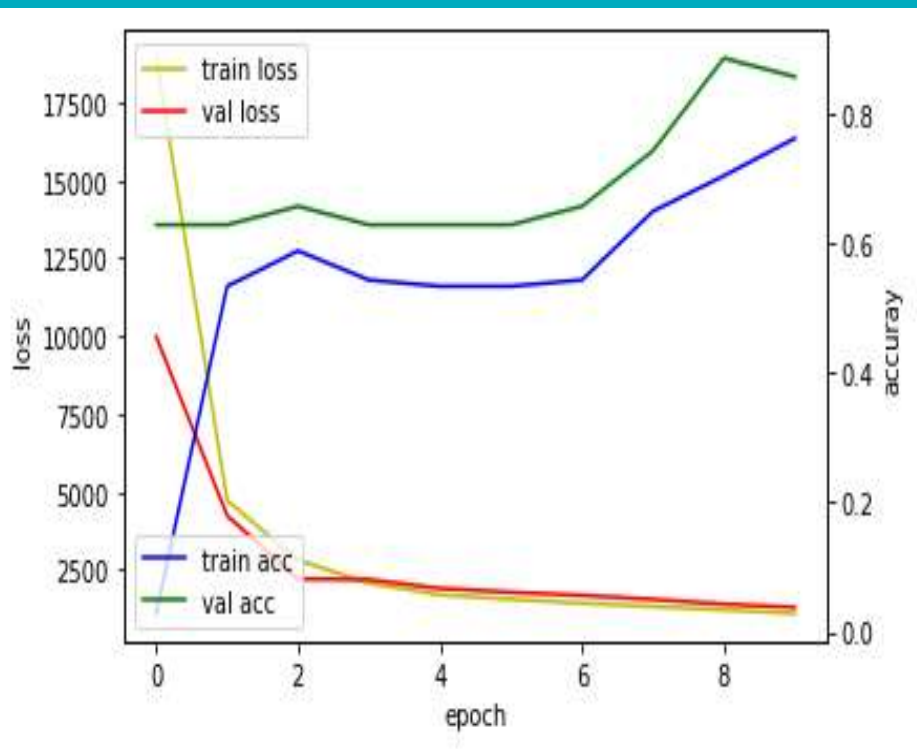
early_stop = keras.callbacks.EarlyStopping(patience=2, restore_best_weights=True)
#과적합 되지않게 early_stop을 경해냈다.

history = model.fit(X_train, y_train_new,
                    validation_data = (X_val, y_val_new),
                    epochs=initial_epochs,
                    batch_size= 32,
                    callbacks=[early_stop])
```

Epoch 1/10
10/10 [=====] - 127s 13s/step - loss: 18916.4277 - accuracy: 0.0289 - val_loss: 9986.1826 - val_accuracy: 0.6286
Epoch 2/10
10/10 [=====] - 135s 14s/step - loss: 4728.9194 - accuracy: 0.5338 - val_loss: 4241.8325 - val_accuracy: 0.6286
Epoch 3/10
10/10 [=====] - 148s 15s/step - loss: 2812.0767 - accuracy: 0.5884 - val_loss: 2208.9651 - val_accuracy: 0.6571
Epoch 4/10
10/10 [=====] - 152s 15s/step - loss: 2076.0891 - accuracy: 0.5434 - val_loss: 2189.3523 - val_accuracy: 0.6286
Epoch 5/10
10/10 [=====] - 156s 16s/step - loss: 1706.5122 - accuracy: 0.5338 - val_loss: 1904.7141 - val_accuracy: 0.6286
Epoch 6/10
10/10 [=====] - 148s 15s/step - loss: 1560.2980 - accuracy: 0.5338 - val_loss: 1784.1740 - val_accuracy: 0.6286
Epoch 7/10
10/10 [=====] - 162s 17s/step - loss: 1434.3702 - accuracy: 0.5434 - val_loss: 1683.2074 - val_accuracy: 0.6571
Epoch 8/10
10/10 [=====] - 165s 16s/step - loss: 1326.5802 - accuracy: 0.6495 - val_loss: 1554.0358 - val_accuracy: 0.7429
Epoch 9/10
10/10 [=====] - 169s 17s/step - loss: 1220.2050 - accuracy: 0.7042 - val_loss: 1401.4913 - val_accuracy: 0.8857
Epoch 10/10
10/10 [=====] - 161s 16s/step - loss: 1107.4655 - accuracy: 0.7621 - val_loss: 1299.6473 - val_accuracy: 0.8571

*과적합 : 학습데이터에 대해서는 오차가 감소하지만 실제 데이터에 대해서는 오차가 증가하게 된다.

손실도와 정확도



*Train loss : 훈련 손실

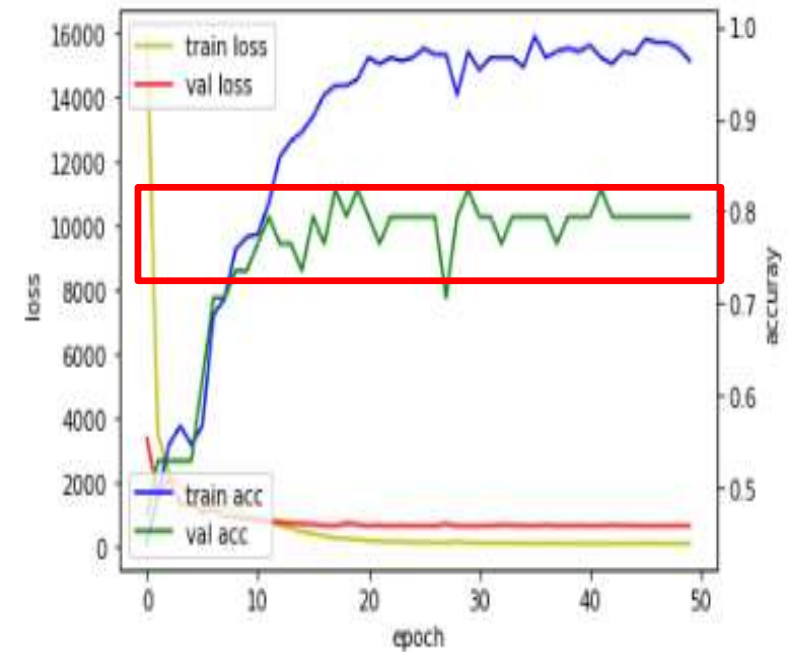
*Val loss : 검증 손실

*Train acc: 훈련 정확도

*Val acc: 검증 정확도

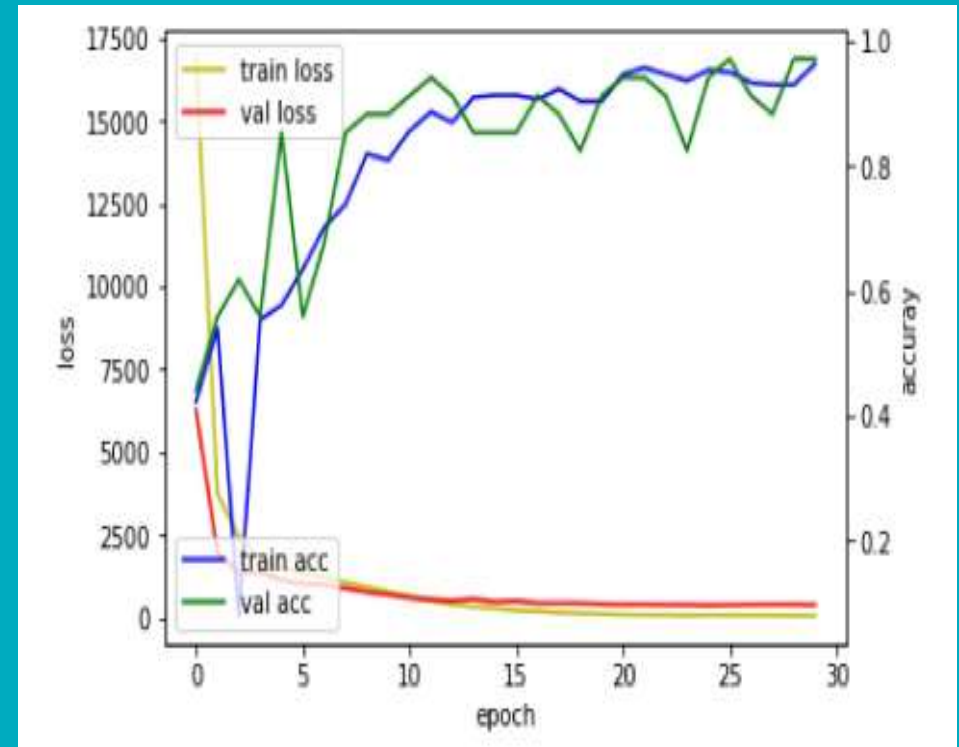
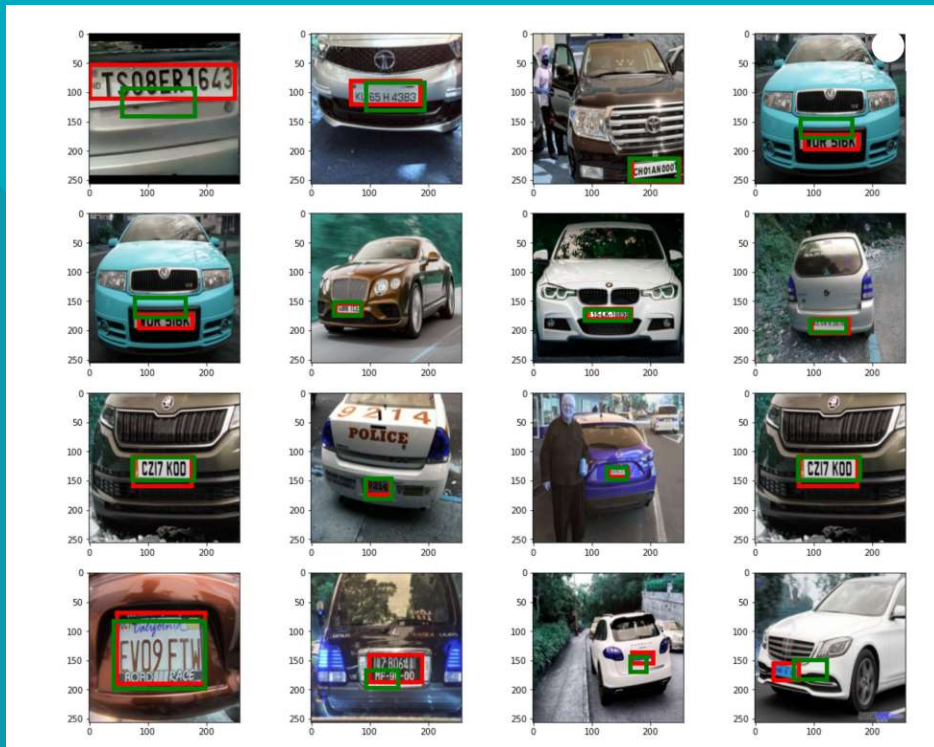
손실도와 정확도

```
Epoch 46/50  
10/10 [=====] - 43s 4s/step - loss: 81.0343 - accuracy: 0.9886 - val_loss: 620.1758 - val_a  
ccuracy: 0.7941  
Epoch 47/50  
10/10 [=====] - 42s 4s/step - loss: 77.4748 - accuracy: 0.9832 - val_loss: 627.5321 - val_a  
ccuracy: 0.7941  
Epoch 48/50  
10/10 [=====] - 42s 4s/step - loss: 69.8501 - accuracy: 0.9832 - val_loss: 628.1489 - val_a  
ccuracy: 0.7941  
Epoch 49/50  
10/10 [=====] - 43s 4s/step - loss: 74.4166 - accuracy: 0.9765 - val_loss: 631.1550 - val_a  
ccuracy: 0.7941  
Epoch 50/50  
10/10 [=====] - 42s 4s/step - loss: 77.1498 - accuracy: 0.9631 - val_loss: 619.7396 - val_a  
ccuracy: 0.7941
```



*Early stop 주석 처리 후 실행

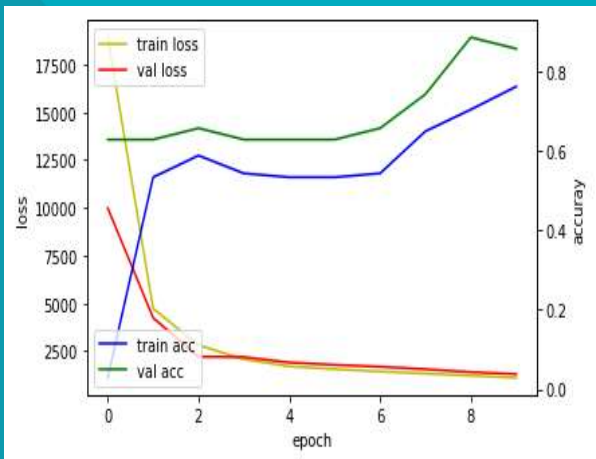
손실도와 정확도



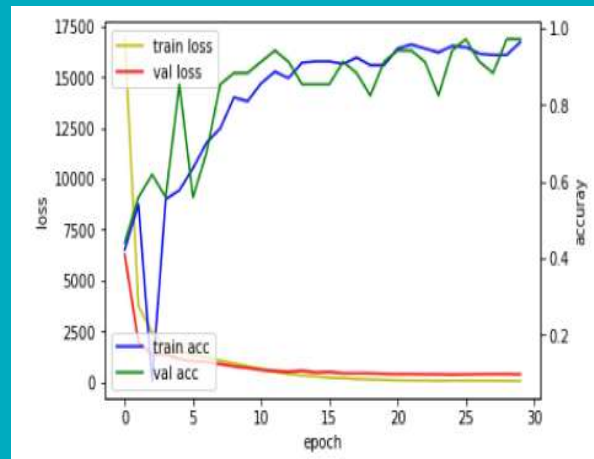
*30번의 학습

손실도와 정확도

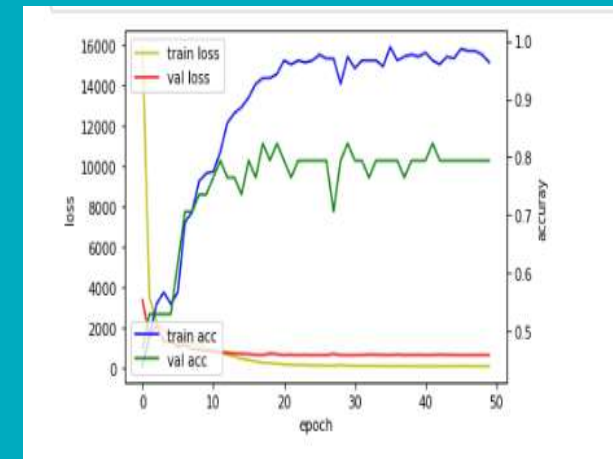
10번의 학습



30번의 학습

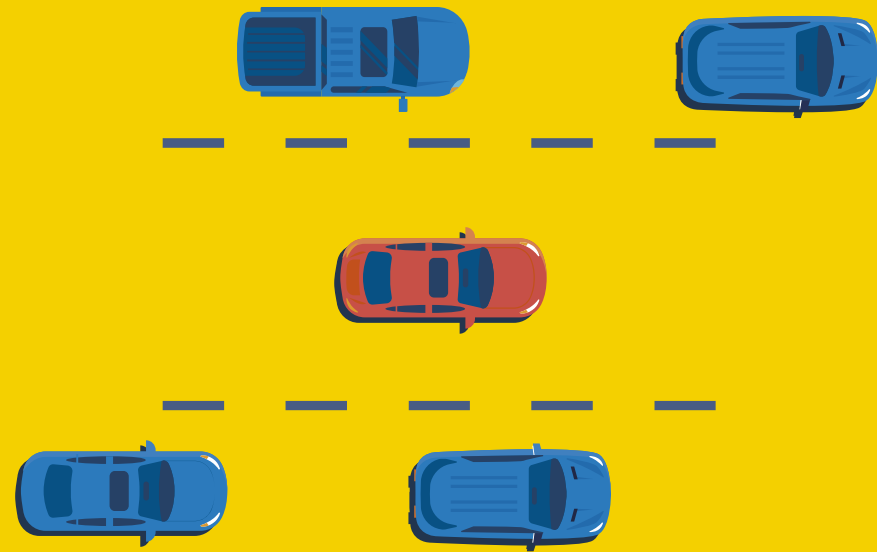


50번의 학습




04

시연 및 마무리



시연 동영상

```
jupyter car-plate-test Last Checkpoint: 어제 오후 1:21 (untrusted) www.BANDICAM.com  Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

In [1]: import numpy as np
import pandas as pd
import cv2

from bs4 import BeautifulSoup
boxes = {}
coches = {}
import os
# 필요한 라이브러리 임포트
for dirname, _, filenames in os.walk('C:\\Users\\larb\\Downloads\\archive'):
    for filename in filenames:
        if filename[-3:] == ".xml":
            xml = open(os.path.join(dirname, filename), "r")
            contents = xml.read()
            soup = BeautifulSoup(contents, 'xml')
            xmin = soup.find('xmin').text
            ymin = soup.find('ymin').text
            xmax = soup.find('xmax').text
            ymax = soup.find('ymax').text
            boxes[filename.split('/')[1][0:-4]] = (xmin,ymin,xmax,ymax)
        else:
            img_org = cv2.imread(os.path.join(dirname, filename))
            #print(img_org.shape)
            coches[filename.split('/')[1][0:-4]] = img_org
#xml 정해져있는 모범답안 - 뷰티풀소프로 미리 태그되어있는 좌표값을 가져옴
# 데이터셋 임포트하는 과정

In [2]: X = []
y = []
import matplotlib.pyplot as plt
for car in coches:
    data_coche = coches[car]
    data_coche_resized = cv2.resize(data_coche, (256,256))
    y_ = data_coche.shape[0]
    x_ = data_coche.shape[1]
    x_scale = (256/x_)
    y_scale = (256/y_)
    box_car = boxes[car]
```


팀원별 소감

장정우

첫 프로젝트라서 많이 미숙했지만 팀원의 격려와 조언 덕분에 역할을 충분히 수행하였습니다. 팀원과 아이디어 회의를 하며 자동차 번호판 인식 아이디어를 얻었고, 교수님의 수업을 토대로 신경망을 참고하여 만들었습니다. 효율을 알아보는 코드와, 학습 과정에서 많은 시행 착오를 겪었지만 포기하지 않고 끝까지 한 결과 저의 첫 프로젝트를 잘 수행하게 된 것 같습니다. 앞으로도 교수님의 좋은 수업을 통해 많이 배우겠습니다. 감사합니다.

김규영

이번 프로젝트 과제를 통해 딥러닝의 개념과 다양한 학습 모델 중 신경망 모델을 통해 자동차 번호판을 인식 딥러닝 코드를 작성하는 과정을 통해 저의 부족한 부분을 알게 되고 부족한 점을 팀원을 통해 보다 수월하게 채워나갈 수 있었습니다. 최종 프로젝트를 팀으로 진행한 덕에 협업하는 법과 과목에 대한 지식을 습득하는 유익한 시간이 되었습니다.



THANKS

대한민국 모든 도로가
안전하길 바랍니다.
감사합니다.

