Second Programming Exam.
Deep Learning, 2022

姓名：吳佳展　學號：110318517

```
import tensorflow as tf
import tensorflow.keras as keras
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import Input, Dense, Dropout, Activation
from tensorflow.keras.layers import BatchNormalization, Flatten
from tensorflow.keras.layers import Conv2D, MaxPooling2D
import pandas as pd
import numpy as np
from sklearn import preprocessing
import matplotlib.pyplot as plt


def Plot_loss(X, y, mode):
    plt.figure(figsize=(12,6))
    plt.plot(X)
    plt.plot(y)
    if mode =='loss':
        plt.title('Model loss')
        plt.ylabel('loss')
        plt.xlabel('epoch')
        plt.legend(['Train_loss', 'Val_loss'], loc='upper right')
    else:
        plt.title('Model Accuracy')
        plt.ylabel('Accuracy')
        plt.xlabel('epoch')
        plt.legend(['Train_ACC', 'Val_ACC'], loc='lower right')

def Predict():
    df_test = pd.read_csv('class_tst.csv',header=None)
    X_data = df_test.values
    X_data_normalized = preprocessing.scale(X_data)
    ans = fnn_model.predict(X_data_normalized)
    fin_ans = ans.argmax(-1) + 1
    result = pd.concat([df_test,pd.DataFrame(fin_ans)],axis=1)
    return result
```

```
def Export_csv(data):
    cols= [i for i in range(0,9)]
    df_ans.to_csv('Q1/class_answer.csv',header=cols,index=None)


# Get data
df = pd.read_csv('class_trn.csv',header=None)
X = df.iloc[:,0:8].values
y = df.iloc[:,8].values

# Normalize data
X_normalized = preprocessing.scale(X)

# split train(80), test, val data(20)
train_len, test_len = int(0.8*len(df)), int(0.2*len(df))
X_train, y_train = X_normalized[:train_len], y[:train_len]
X_test, y_test = X_normalized[train_len:], y[train_len:]
y_train, y_test = y_train-1, y_test - 1
y_train = y_train.reshape(len(y_train),1)
y_test = y_test.reshape(len(y_test),1)
y_train = np.eye(5, dtype='float32')[y_train[:, 0]]
y_test = np.eye(5, dtype='float32')[y_test[:, 0]]

# Create model
inputs = Input(shape=X_train.shape[1:])
x = Dense(40, activation='relu')(inputs)
x = Dense(35, activation='relu')(x)
x = Dense(20, activation='relu')(x)
x = Dense(20, activation='relu')(x)
x = Dense(40, activation='relu')(x)
x = Flatten()(x)
outputs = Dense(5, activation='softmax')(x)
fnn_model = Model(inputs=inputs, outputs=outputs)
fnn_model.summary()

# Training model
learning_rate = 1e-4
```

```
optimizer = keras.optimizers.Adam(learning_rate=learning_rate)
fnn_model.compile(loss='categorical_crossentropy', optimizer=optimizer,
metrics=['accuracy'])
history = fnn_model.fit(X_train, y_train, batch_size=32, epochs=100,
validation_data=(X_test, y_test), verbose=2)
fnn_model.save('Q1/class.h5')

# Plot loss, accuracy picture
Plot_loss(history.history['loss'], history.history['val_loss'], 'loss')
Plot_loss(history.history['accuracy'], history.history['val_accuracy'], 'acc')

# Use model predict
df_ans = Predict()

# Export csv
Export_csv(df_ans)
```