```python
# -*- coding: utf-8 -*-
"""TensorFlow_Model.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1io3Hj9bFn56RH23DcKvgA2545zUIGxoc

<center><h1>Mini Project 3 - Convolutional Neural Network</h1>
<h4>The TensorFlow File.</h4></center>

<h3>Team Members:</h3>
<center>
Yi Zhu, 260716006<br>
Fei Peng, 260712440<br>
Yukai Zhang, 260710915
</center>
"""

import torch
import torchvision
import tensorflow as tf
from sklearn.model_selection import train_test_split
import time
import pickle
import numpy as np
import pandas as pd
from PIL import Image
import torchvision.transforms as transforms
import matplotlib.pyplot as plt

from torch.utils.data import Dataset
from torch.utils.data import DataLoader

import tensorflow.keras as keras
from tensorflow.keras.applications.vgg19 import VGG19
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D, Dropout
from tensorflow.keras.optimizers import SGD

# Commented out IPython magic to ensure Python compatibility.
from google.colab import drive
drive.mount("/content/drive")

# %cd '/content/drive/My Drive/ECSE_551_F_2020/Mini_Project_03/'

"""# Import and Preprocess Dataset"""

class MyDataset(Dataset):
    def __init__(self, img_file, label_file, transform=None, idx = None):
        self.data = pickle.load( open( img_file, 'rb' ), encoding='bytes')
        self.targets = np.genfromtxt(label_file, delimiter=',', skip_header=1
)[:,1:]
        if idx is not None:
          self.targets = self.targets[idx]
          self.data = self.data[idx]
        self.transform = transform
        self.targets -= 5

    def __len__(self):
        return len(self.targets)

    def __getitem__(self, index):
        img, target = self.data[index], int(self.targets[index])
        img = Image.fromarray(img.astype('uint8'), mode='L')

        if self.transform is not None:
           img = self.transform(img)

        return img, target

img_transform = transforms.Compose([
    transforms.ToTensor(),
    transforms.Normalize((0.5,), (0.5,))
])

batch_size = 32 #feel free to change it
# Read image data and their label into a Dataset class
train_set = MyDataset('./Train.pkl', './TrainLabels.csv', transform=
img_transform, idx=None)

train_set_data = np.repeat(train_set.data[..., np.newaxis], 3, -1)
# train_x, x_validation, train_y, y_validation = train_test_split(
train_set_data, train_set.targets, test_size=0.20, random_state=0)
train_x = train_set_data
train_y = train_set.targets
print(train_x.shape, train_y.shape)

"""# Define Model"""

vgg19 = VGG19(weights=None, include_top=False)

# add a global spatial average pooling layer
x = vgg19.output
x = GlobalAveragePooling2D()(x)
x = Dropout(0.3)(x)
x = Dense(128, activation='relu')(x)
predictions = Dense(9, activation='softmax')(x)

# this is the model to train
model = Model(inputs=vgg19.input, outputs=predictions)

# make all layers trainable
for layer in vgg19.layers:
    layer.trainable = True

"""# Train"""

model.compile(keras.optimizers.SGD(learning_rate=0.001, momentum=0.7), loss='
sparse_categorical_crossentropy', metrics=['accuracy'])

# model.fit(train_x, train_y, epochs=45, validation_data=(x_validation,
y_validation), batch_size=32)
model.fit(train_x, train_y, epochs=65, batch_size=32)

"""# Output Export"""

test_data = pickle.load(open( './Test.pkl', 'rb' ), encoding='bytes')
test_data = np.repeat(test_data[..., np.newaxis], 3, -1)

from google.colab import files
import pandas as pd

y_test = model.predict(test_data, batch_size=32)
X_id = np.arange(y_test.shape[0])

```

```
y_class = np.argmax(y_test,axis=1) + 5
print(y_class)
result = {'id': X_id, 'class': y_class}

df = pd.DataFrame(data=result)
df.to_csv('result.csv', index=False)
files.download('result.csv')
```