import sys, os, multiprocessing, csv, random, time

import urllib.request as request

from PIL import Image

from io import StringIO, BytesIO

import numpy as np

def ParseData(data\_file):

csvfile = open(data\_file, 'r')

csvreader = csv.reader(csvfile)

key\_url\_list = [line[:2] for line in csvreader]

return key\_url\_list[1:] # Chop off header

def DownloadImage(key\_url, out\_dir='data'):

(key, url) = key\_url

filename = os.path.join(out\_dir, '%s.jpg' % key)

if os.path.exists(filename):

print('Image %s already exists. Skipping download.' % filename)

return

try:

response = request.urlopen(url)

image\_data = response.read()

print(image\_data)

except:

print('Warning: Could not download image %s from %s' % (key, url))

return

try:

pil\_image = Image.open(BytesIO(image\_data))

except:

print('Warning: Failed to parse image %s' % key)

return

try:

pil\_image\_rgb = pil\_image.convert('RGB')

except:

print('Warning: Failed to convert image %s to RGB' % key)

return

try:

pil\_image\_rgb.save(filename, format='JPEG', quality=90)

except:

print('Warning: Failed to save image %s' % filename)

return

# purpose: download a list of images

# parameters:

# key\_url\_list: the list of tuples (key, url)

# type: str

# out\_dir: the path to the direction where downloaded images are stored

def DownloadMultipleImage(key\_url\_list, out\_dir, n\_processes=1):

if not os.path.exists(out\_dir):

os.mkdir(out\_dir)

pool = multiprocessing.Pool(processes=n\_processes)

pool.map(DownloadImage, key\_url\_list)

if \_\_name\_\_ == '\_\_main\_\_':

\_\_spec\_\_ = "ModuleSpec(name='builtins', loader=<class '\_frozen\_importlib.BuiltinImporter'>)"

data\_file = 'train.csv'

out\_dir = 'data'

# read keys & urls from data file

key\_url\_list = ParseData(data\_file)

# sample n urls

random.shuffle(key\_url\_list)

n\_samples = 5

chosen\_idxs = np.random.choice(np.arange(len(key\_url\_list)), size=n\_samples, replace=False)

chosen\_list = key\_url\_list[:n\_samples]

# read images

DownloadMultipleImage(chosen\_list, out\_dir, 50)