HW 4 Мельчук А.Б.

Обучить модель семантической сегментации (человек-vs-фон) на подмножестве датасета MS COCO
Библиотеки: [Python, Tensorflow]

Переключение версии TensorFlow

In [1]:

```
import os
import skimage.io as io
import numpy as np

import tensorflow as tf
import matplotlib.pyplot as plt
```

In [2]:

```
physical_devices = tf.config.experimental.list_physical_devices('GPU')
for physical_device in physical_devices:
    tf.config.experimental.set_memory_growth(physical_device, True)
```

Подготовка COCO API

In [4]:

```
COCO_ROOT = '/Volumes/SSD_EVO850PRO/COCO/'
import sys
sys.path.insert(0, os.path.join(COCO_ROOT, 'cocoapi/PythonAPI'))
from pycocotools.coco import COCO
```

In [5]:

```
class Dataset():

    def crop_images(self, img, inp_size, random_crop=False):
        shape = tf.shape(img)
        pad = (
            [0, tf.maximum(inp_size - shape[0], 0)],
            [0, tf.maximum(inp_size - shape[1], 0)],
            [0, 0],
        )
        img = tf.pad(img, pad)

        if random_crop:
            img = tf.image.random_crop(img, (inp_size, inp_size, shape[2]))
        else: # central crop
            shape = tf.shape(img)
            ho = (shape[0] - inp_size) // 2
            wo = (shape[1] - inp_size) // 2
            img = img[ho:ho+inp_size, wo:wo+inp_size, :]

        return img

    def train_dataset(self, batch_size, epochs, inp_size):

        def item_to_images(item):
            random_crop = True
            img_combined = tf.py_function(self.read_images, [item], tf.uint8)
            img_combined = self.crop_images(img_combined, inp_size, random_crop)

            img = tf.cast(img_combined[...,:3], tf.float32) / np.float32(255.)
            mask_class = tf.cast(img_combined[...,3:4], tf.float32)
            return img, mask_class

        dataset = tf.data.Dataset.from_tensor_slices(self.img_list)
        dataset = dataset.shuffle(buffer_size=len(self.img_list))
        dataset = dataset.map(item_to_images)
        dataset = dataset.repeat(epochs)
        dataset = dataset.batch(batch_size, drop_remainder=True)

        return dataset

    def val_dataset(self, batch_size, inp_size):

        def item_to_images(item):
            random_crop = False
            img_combined = tf.py_function(self.read_images, [item], tf.uint8)
            img_combined = self.crop_images(img_combined, inp_size, random_crop)

            img = tf.cast(img_combined[...,:3], tf.float32) / np.float32(255.)
            mask_class = tf.cast(img_combined[...,3:4], tf.float32)
            return img, mask_class

        dataset = tf.data.Dataset.from_tensor_slices(self.img_list)
        dataset = dataset.map(item_to_images)
        dataset = dataset.batch(batch_size, drop_remainder=True)

        return dataset
```

In [6]:

```
class COCO_Dataset(Dataset):

    def __init__(self, sublist):
        ann_file_fpath = os.path.join(COCO_ROOT, 'annotations', 'instances_'+sublist
        self.coco = COCO(ann_file_fpath)
        self.cat_ids = self.coco.getCatIds(catNms=['person'])
        self.img_list = self.coco.getImgIds(catIds=self.cat_ids)

    def read_images(self, img_id):
        img_id = int(img_id.numpy())
        img_data = self.coco.loadImgs(img_id)[0]
        img_fname = '/'.join(img_data['coco_url'].split('/')[-2:])

        img = io.imread(os.path.join(COCO_ROOT, img_fname))
        if len(img.shape) == 2:
            img = np.tile(img[..., None], (1, 1, 3))

        ann_ids = self.coco.getAnnIds(imgIds=img_data['id'], catIds=self.cat_ids, is
        anns = self.coco.loadAnns(ann_ids)
        mask_class = np.zeros((img.shape[0], img.shape[1]), dtype=np.uint8)
        for i in range(len(anns)):
            mask_class += self.coco.annToMask(anns[i])
        mask_class = (mask_class > 0).astype(np.uint8)

        img_combined = np.concatenate([img, mask_class[..., None]], axis=2)

        return img_combined
```

In [7]:

```
COCO_dataset_train = COCO_Dataset('train')
COCO_dataset_val = COCO_Dataset('val')
```

```
loading annotations into memory...
Done (t=18.72s)
creating index...
index created!
loading annotations into memory...
Done (t=0.64s)
creating index...
index created!
```

In [8]:

```
train_ds = COCO_dataset_train.train_dataset(batch_size=8, epochs=1, inp_size=256)
val_ds = COCO_dataset_val.val_dataset(batch_size=8, inp_size=256)
```

Модель ASPP

In [9]:

```
class ASPPBlock(tf.keras.Model):
    def __init__(self):
        super().__init__()
        self.conv1 = tf.keras.layers.Conv2D(256, (1, 1), padding='same', activation=
        self.conv2 = tf.keras.layers.Conv2D(256, (3, 3), dilation_rate=6, padding='s
        self.conv3 = tf.keras.layers.Conv2D(256, (3, 3), dilation_rate=12, padding='
        self.conv4 = tf.keras.layers.Conv2D(256, (3, 3), dilation_rate=18, padding='
        self.conv5 = tf.keras.layers.Conv2D(256, (1, 1), padding='same', activation=

    def call(self, inp, is_training=False):
        out1 = self.conv1(inp)
        out2 = self.conv2(inp)
        out3 = self.conv3(inp)
        out4 = self.conv4(inp)
        out = tf.concat([out1, out2, out3, out4], axis=3)
        out = self.conv5(out)
        return out
```

Создание модели U-Net

In [10]:

```
def build_model():
    img_input = tf.keras.layers.Input((256, 256, 3))

    x = tf.keras.layers.Conv2D(64, (3, 3), activation='relu', padding='same', name='
    x = tf.keras.layers.Conv2D(64, (3, 3), activation='relu', padding='same', name='
    x = tf.keras.layers.MaxPooling2D((2, 2), strides=(2, 2), name='unit1_pool')(x)

    x = tf.keras.layers.Conv2D(128, (3, 3), activation='relu', padding='same',name='
    x = tf.keras.layers.Conv2D(128, (3, 3), activation='relu', padding='same',name='
    x = tf.keras.layers.MaxPooling2D((2, 2), strides=(2, 2), name='unit2_pool')(x)

    x = tf.keras.layers.Conv2D(256, (3, 3), activation='relu', padding='same',name='
    x = tf.keras.layers.Conv2D(256, (3, 3), activation='relu', padding='same',name='
    x = tf.keras.layers.Conv2D(256, (3, 3), activation='relu', padding='same',name='
    out_enc_mid = x
    x = tf.keras.layers.MaxPooling2D((2, 2), strides=(2, 2), name='unit3_pool')(x)

    x = tf.keras.layers.Conv2D(512, (3, 3), activation='relu', padding='same',name='
    x = tf.keras.layers.Conv2D(512, (3, 3), activation='relu', padding='same',name='
    x = tf.keras.layers.Conv2D(512, (3, 3), activation='relu', padding='same',name='
    x = tf.keras.layers.MaxPooling2D((2, 2), strides=(2, 2), name='unit4_pool')(x)

    x = tf.keras.layers.Conv2D(512, (3, 3), activation='relu', padding='same',name='
    x = tf.keras.layers.Conv2D(512, (3, 3), activation='relu', padding='same',name='
    x = tf.keras.layers.Conv2D(512, (3, 3), activation='relu', padding='same',name='

    x = aspp(x)

    x = tf.image.resize(x, tf.shape(out_enc_mid)[1:3], tf.image.ResizeMethod.BILINEA

    out_enc_mid = tf.keras.layers.Conv2D(48, (1, 1), padding='same', activation='rel

    x = tf.concat([x, out_enc_mid], axis=3)

    x = tf.keras.layers.Conv2D(256, (3, 3), padding='same', activation='relu')(x)
    x = tf.keras.layers.Conv2D(256, (3, 3), padding='same', activation='relu')(x)
    x = tf.keras.layers.Conv2D(1, (1, 1), padding='same', activation=None)(x)

    x = tf.image.resize(x, tf.shape(img_input)[1:3], tf.image.ResizeMethod.BILINEAR)
    x = tf.nn.sigmoid(x)

    return tf.keras.Model(inputs=img_input, outputs=x)

aspp = ASPPBlock()
model = build_model()
```

In [11]:

```
model.load_weights(COCO_ROOT + 'weights.h5', by_name=True)
```

In [12]:

```
loss = tf.keras.losses.BinaryCrossentropy()
model.compile(optimizer='adam', loss=loss)
```

In [13]:

```
model.fit(train_ds, validation_data=val_ds)
```

```
Train for 8014 steps, validate for 336 steps
 124/8014 [..............................] - ETA: 3:49:14 - loss: 0.7
516WARNING:tensorflow:Can save best model only with val_acc availabl
e, skipping.
 249/8014 [..............................] - ETA: 3:38:58 - loss: 0.6
361WARNING:tensorflow:Can save best model only with val_acc availabl
e, skipping.
 374/8014 [>.............................] - ETA: 3:33:12 - loss: 0.5
823WARNING:tensorflow:Can save best model only with val_acc availabl
e, skipping.
 499/8014 [>.............................] - ETA: 3:28:34 - loss: 0.5
497WARNING:tensorflow:Can save best model only with val_acc availabl
e, skipping.
 624/8014 [=>............................] - ETA: 3:23:13 - loss: 0.5
505WARNING:tensorflow:Can save best model only with val_acc availabl
e, skipping.
 749/8014 [=>............................] - ETA: 3:18:33 - loss: 0.5
390WARNING:tensorflow:Can save best model only with val_acc availabl
e, skipping.
```

In [14]:

```
train_x = [x[0] for x in iter(train_ds.take(20))]
for sample in train_x:
    out = model.predict(sample[0][None, ...])
    seg_map = (out[0, ..., 0]>0.5).astype(np.float32)
    seg_map_clr = plt.get_cmap('viridis')(seg_map)[..., :3]
    plt.imshow(sample[0]*0.5 + seg_map_clr*0.5)
    plt.show()
```