```
import os
import glob
import random
import numpy as np
import matplotlib.pyplot as plt

import tensorflow as tf
```

```
if 1:
    !pip install scikit-video==1.1.11
import skvideo.io
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/
Requirement already satisfied: scikit-video==1.1.11 in /usr/local/lib/python3.7/dist
Requirement already satisfied: pillow in /usr/local/lib/python3.7/dist-packages (fro
Requirement already satisfied: scipy in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages (from
```

```
classes = [
    # 'jogging',
    'running',
    'boxing',
    'walking',
    # 'handwaving',
```

Ошибка автоматического сохранения. Этот файл был обновлен удаленно или на другой вкладке. Показать различия

```
dataset = []
data_root = './'
for cls in classes:
    print('Processing class: {}'.format(cls))
    for fpath in glob.glob(os.path.join(data_root, cls, '*.avi')):
        cls_idx = classes.index(cls)
        dataset.append((fpath, cls_idx))
```

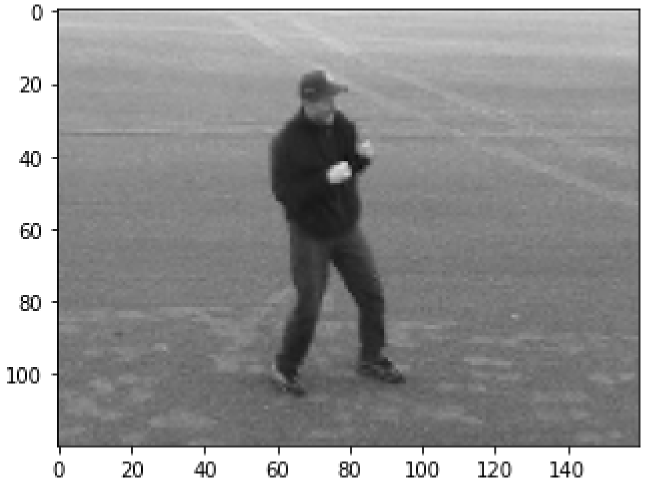
```
Processing class: running
Processing class: boxing
Processing class: walking
```

```
SUBSET_LEN = 10
random.shuffle(dataset)
dataset = dataset[:SUBSET_LEN]

print('Dataset samples (subset):', len(dataset))
```

```
Dataset samples (subset): 10
```

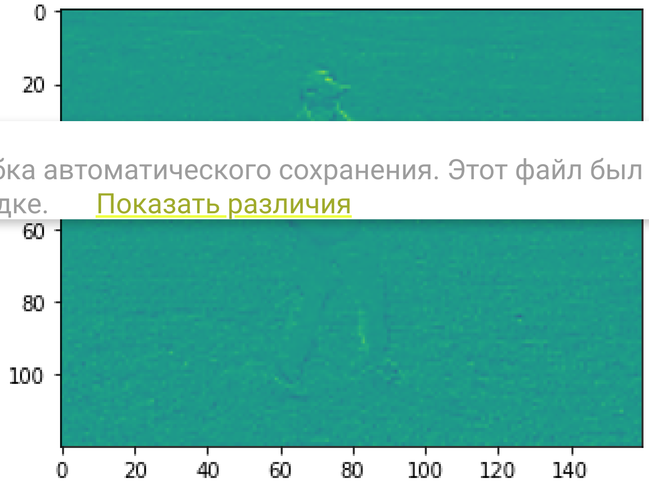
```
videodata = skvideo.io.vread(dataset[0][0])
videodata = videodata.astype(np.float32) / 255.
print('videodata shape:', videodata.shape)
plt.imshow(videodata[50, ...])
plt.imshow(videodata[50, ...])
```

```
videodata shape: (482, 120, 160, 3)
<matplotlib.image.AxesImage at 0x7efcb120bed0>
```



```
motion = np.mean(videodata[1:, ...] - videodata[:-1, ...], axis=3, keepdims=True)
print('motion shape:', motion.shape)
plt.imshow(motion[50, ..., 0])
```

```
motion shape: (481, 120, 160, 1)
<matplotlib.image.AxesImage at 0x7efcaf493450>
```



Ошибка автоматического сохранения. Этот файл был обновлен удаленно или на другой вкладке. Показать различия

```
model = tf.keras.Sequential([
    tf.keras.layers.Conv3D(32, (5, 5, 5), (1, 2, 2), padding='same', activation='relu'),
    tf.keras.layers.MaxPool3D((1, 2, 2), padding='same'),
    tf.keras.layers.Conv3D(64, (5, 5, 5), (1, 2, 2), padding='same', activation='relu'),
    tf.keras.layers.MaxPool3D((1, 2, 2), padding='same'),
    tf.keras.layers.Conv3D(64, (3, 3, 3), (1, 2, 2), padding='same', activation='relu'),
    tf.keras.layers.MaxPool3D((1, 2, 2), padding='same'),
    tf.keras.layers.Conv3D(64, (3, 3, 3), (1, 1, 1), padding='same', activation=None),
    tf.keras.layers.GlobalAveragePooling3D(),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(6, activation=None),
])
```

```
inp = motion[None, ...]
out = model(inp)
```

```
print('Input shape:', inp.shape)
print('Output shape:', out.shape)
```

```
     Input shape: (1, 481, 120, 160, 1)
     Output shape: (1, 6)
```

```
NUM_EPOCHS = 10
LEARNING_RATE = 0.001

model.compile(
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
    optimizer=tf.keras.optimizers.Adam(LEARNING_RATE))

writer = tf.summary.create_file_writer('logs/exp1')
```

```
global_step = 0
for ep in range(NUM_EPOCHS):
    for iter, (fpath, label) in enumerate(dataset):
        videodata = skvideo.io.vread(fpath)
        videodata = videodata.astype(np.float32) / 255.
        motion = np.mean(videodata[1:, ...] - videodata[:-1, ...], axis=3, keepdims=True)
        x = motion[None, ...]
        y = np.array(label)[None, ...]

        loss_value = model.train_on_batch(x, y)
```

Ошибка автоматического сохранения. Этот файл был обновлен удаленно или на другой вкладке. Показать различия

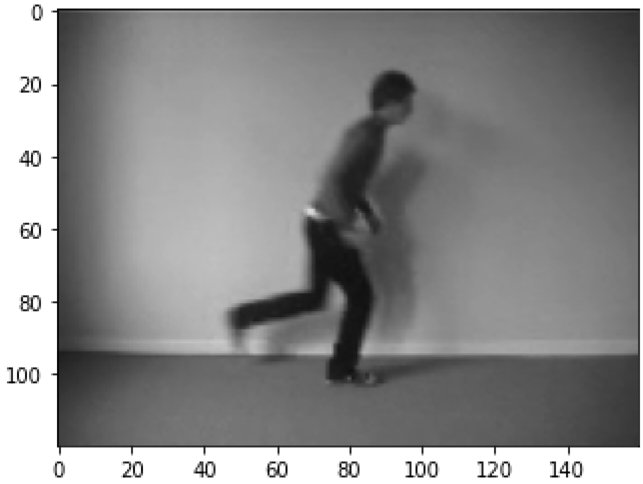
```
            print(f'[{ep}/{NUM_EPOCHS}][{iter}/{len(dataset)}] Loss = {loss_value}')

            with writer.as_default():
                tf.summary.scalar('loss', loss_value, global_step)

        global_step += 1
```

```
     [0/10][0/10] Loss = 1.787575125694275
     [1/10][0/10] Loss = 1.5494177341461182
     [2/10][0/10] Loss = 1.5620465278625488
     [3/10][0/10] Loss = 1.0333774089813232
     [4/10][0/10] Loss = 1.216446876525879
     [5/10][0/10] Loss = 0.5812814831733704
     [6/10][0/10] Loss = 0.11040806025266647
     [7/10][0/10] Loss = 0.0013466347008943558
     [8/10][0/10] Loss = 0.004491479601711035
     [9/10][0/10] Loss = 0.0021219374611973763
```

```
fpath, cls_true = random.choice(dataset)

videodata = skvideo.io.vread(fpath)
videodata = videodata.astype(np.float32) / 255.
plt.imshow(videodata[30, ...])

motion = np.mean(videodata[1:, ...] - videodata[:-1, ...], axis=3, keepdims=True)
```

```
out = model(motion[None, ...])[0]
cls_pred = np.argmax(out.numpy())

print('True class:', classes[cls_true])
print('Predicted class:', classes[cls_pred])
```

```
True class: running
Predicted class: running
```



Ошибка автоматического сохранения. Этот файл был обновлен удаленно или на другой вкладке. Показать различия

Платные продукты Colab - Отменить подписку

✓ 3 сек. выполнено в 04:41 ● ✕