МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ИМ. Р.Е. АЛЕКСЕЕВА
ИНСТИТУТ РАДИОЭЛЕКТРОНИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Курс “Аппаратное и программное обеспечение роботизированных систем”

Отчет по лабораторной работе №3

Выполнил: Соков С.А.
Проверил: Гай В.Е.

Нижний Новгород 2021

Тема работы:

Классификация изображений с использованием свёрточных нейронных сетей.

Задание:

Выполнить анализ статьи, разобрать структуру сети, реализовать сеть в Keras, оценить точность работы сети.

Вариантданных: MNIST

Вариант модели сети: Xception

Листинг программы:

```
[ ] import keras
    import tensorflow as tf

    from keras.applications.xception import Xception, preprocess_input
    from keras.models import Sequential, Model
    from keras.layers import Lambda, Input
    from keras.backend import tf as ktf
    from keras.layers.core import Flatten, Dense, Dropout
    from keras.utils import np_utils
    from keras.optimizers import SGD,Adam
    from scipy import misc
    import numpy as np

    mnist = tf.keras.datasets.mnist
```

```
[ ] (x_train, y_train),(x_test, y_test) = mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11493376/11490434 [==============================] - 0s 0us/step
```

```
[ ] # Размер изображений
    img_width, img_height = 28, 28
    # Размер мини-выборки
    batch_size = 32
    # Кол-во изображений для обучения
    nb_train_samples = 60000
    # Кол-во изображений для теста
    nb_test_samples = 10000
```

```
[ ] # Нормалтзация изображений к значениям -1, 1
    x_train = tf.keras.utils.normalize(x_train, axis=1)
    x_test = tf.keras.utils.normalize(x_test, axis=1)

    x_train = x_train.astype('float32')
    x_test = x_test.astype('float32')
    x_train = x_train / 255.0
    x_test = x_test / 255.0
    x_train = np.stack((x_train,)*3, axis=-1)
    x_test = np.stack((x_test,)*3, axis=-1)

    y_train = np_utils.to_categorical(y_train)
    y_test = np_utils.to_categorical(y_test)
    num_classes = y_test.shape[1]
```

```
# Создание экземпляра модели сети
xception = tf.keras.applications.Xception(
    include_top=False,
    weights="imagenet",
    input_shape= (128,128,3)
)
# input_shape : Необязательный кортеж формы
# include_top : логическое значение, следует ли включать полностью
#подключенный уровень в верхнюю часть сети.
# weights : один из None(случайная инициализация), «imagenet»
#(предварительное обучение в ImageNet) или путь к загружаемому файлу весов.
# Сверточная часть сети
xception.trainable = False
trainable = False
for layer in xception.layers:H
   if layer.name == 'block5_sepconv1':
    trainable = True
   layer.trainable = trainable
# # Сверточную часть сети обучать не надо

xception.summary()
```

Создадим экземпляр модели:

```
Model: "xception"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_13 (InputLayer)           [(None, 128, 128, 3) 0
__________________________________________________________________________________________________
block1_conv1 (Conv2D)           (None, 63, 63, 32)   864         input_13[0][0]
__________________________________________________________________________________________________
block1_conv1_bn (BatchNormaliza (None, 63, 63, 32)   128         block1_conv1[0][0]
__________________________________________________________________________________________________
block1_conv1_act (Activation)   (None, 63, 63, 32)   0           block1_conv1_bn[0][0]
__________________________________________________________________________________________________
block1_conv2 (Conv2D)           (None, 61, 61, 64)   18432       block1_conv1_act[0][0]
__________________________________________________________________________________________________
block1_conv2_bn (BatchNormaliza (None, 61, 61, 64)   256         block1_conv2[0][0]
__________________________________________________________________________________________________
block1_conv2_act (Activation)   (None, 61, 61, 64)   0           block1_conv2_bn[0][0]
__________________________________________________________________________________________________
block2_sepconv1 (SeparableConv2 (None, 61, 61, 128)  8768        block1_conv2_act[0][0]
__________________________________________________________________________________________________
block2_sepconv1_bn (BatchNormal (None, 61, 61, 128)  512         block2_sepconv1[0][0]
__________________________________________________________________________________________________
block2_sepconv2_act (Activation (None, 61, 61, 128)  0           block2_sepconv1_bn[0][0]
__________________________________________________________________________________________________
block2_sepconv2 (SeparableConv2 (None, 61, 61, 128)  17536       block2_sepconv2_act[0][0]
__________________________________________________________________________________________________
block2_sepconv2_bn (BatchNormal (None, 61, 61, 128)  512         block2_sepconv2[0][0]
__________________________________________________________________________________________________
conv2d_48 (Conv2D)              (None, 31, 31, 128)  8192        block1_conv2_act[0][0]
__________________________________________________________________________________________________
block2_pool (MaxPooling2D)      (None, 31, 31, 128)  0           block2_sepconv2_bn[0][0]
__________________________________________________________________________________________________
batch_normalization_48 (BatchNo (None, 31, 31, 128)  512         conv2d_48[0][0]
__________________________________________________________________________________________________
add_144 (Add)                   (None, 31, 31, 128)  0           block2_pool[0][0]
                                                                 batch_normalization_48[0][0]
__________________________________________________________________________________________________
block3_sepconv1_act (Activation (None, 31, 31, 128)  0           add_144[0][0]
```

block3_sepconv1 (SeparableConv2	(None, 31, 31, 256)	33920	block3_sepconv1_act[0][0]
block3_sepconv1_bn (BatchNormal	(None, 31, 31, 256)	1024	block3_sepconv1[0][0]
block3_sepconv2_act (Activation	(None, 31, 31, 256)	0	block3_sepconv1_bn[0][0]
block3_sepconv2 (SeparableConv2	(None, 31, 31, 256)	67840	block3_sepconv2_act[0][0]
block3_sepconv2_bn (BatchNormal	(None, 31, 31, 256)	1024	block3_sepconv2[0][0]
conv2d_49 (Conv2D)	(None, 16, 16, 256)	32768	add_144[0][0]
block3_pool (MaxPooling2D)	(None, 16, 16, 256)	0	block3_sepconv2_bn[0][0]
batch_normalization_49 (BatchNo	(None, 16, 16, 256)	1024	conv2d_49[0][0]
add_145 (Add)	(None, 16, 16, 256)	0	block3_pool[0][0] batch_normalization_49[0][0]
block4_sepconv1_act (Activation	(None, 16, 16, 256)	0	add_145[0][0]
block4_sepconv1 (SeparableConv2	(None, 16, 16, 728)	188672	block4_sepconv1_act[0][0]
block4_sepconv1_bn (BatchNormal	(None, 16, 16, 728)	2912	block4_sepconv1[0][0]
block4_sepconv2_act (Activation	(None, 16, 16, 728)	0	block4_sepconv1_bn[0][0]
block4_sepconv2 (SeparableConv2	(None, 16, 16, 728)	536536	block4_sepconv2_act[0][0]
block4_sepconv2_bn (BatchNormal	(None, 16, 16, 728)	2912	block4_sepconv2[0][0]
conv2d_50 (Conv2D)	(None, 8, 8, 728)	186368	add_145[0][0]
block4_pool (MaxPooling2D)	(None, 8, 8, 728)	0	block4_sepconv2_bn[0][0]
batch_normalization_50 (BatchNo	(None, 8, 8, 728)	2912	conv2d_50[0][0]
add_146 (Add)	(None, 8, 8, 728)	0	block4_pool[0][0] batch_normalization_50[0][0]

block5_sepconv1_act (Activation	(None, 8, 8, 728)	0	add_146[0][0]
block5_sepconv1 (SeparableConv2	(None, 8, 8, 728)	536536	block5_sepconv1_act[0][0]
block5_sepconv1_bn (BatchNormal	(None, 8, 8, 728)	2912	block5_sepconv1[0][0]
block5_sepconv2_act (Activation	(None, 8, 8, 728)	0	block5_sepconv1_bn[0][0]
block5_sepconv2 (SeparableConv2	(None, 8, 8, 728)	536536	block5_sepconv2_act[0][0]
block5_sepconv2_bn (BatchNormal	(None, 8, 8, 728)	2912	block5_sepconv2[0][0]
block5_sepconv3_act (Activation	(None, 8, 8, 728)	0	block5_sepconv2_bn[0][0]
block5_sepconv3 (SeparableConv2	(None, 8, 8, 728)	536536	block5_sepconv3_act[0][0]
block5_sepconv3_bn (BatchNormal	(None, 8, 8, 728)	2912	block5_sepconv3[0][0]
add_147 (Add)	(None, 8, 8, 728)	0	block5_sepconv3_bn[0][0] add_146[0][0]
block6_sepconv1_act (Activation	(None, 8, 8, 728)	0	add_147[0][0]
block6_sepconv1 (SeparableConv2	(None, 8, 8, 728)	536536	block6_sepconv1_act[0][0]
block6_sepconv1_bn (BatchNormal	(None, 8, 8, 728)	2912	block6_sepconv1[0][0]
block6_sepconv2_act (Activation	(None, 8, 8, 728)	0	block6_sepconv1_bn[0][0]
block6_sepconv2 (SeparableConv2	(None, 8, 8, 728)	536536	block6_sepconv2_act[0][0]
block6_sepconv2_bn (BatchNormal	(None, 8, 8, 728)	2912	block6_sepconv2[0][0]
block6_sepconv3_act (Activation	(None, 8, 8, 728)	0	block6_sepconv2_bn[0][0]
block6_sepconv3 (SeparableConv2	(None, 8, 8, 728)	536536	block6_sepconv3_act[0][0]

block13_sepconv1 (SeparableConv	(None, 8, 8, 728)	536536	block13_sepconv1_act[0][0]
block13_sepconv1_bn (BatchNorma	(None, 8, 8, 728)	2912	block13_sepconv1[0][0]
block13_sepconv2_act (Activatio	(None, 8, 8, 728)	0	block13_sepconv1_bn[0][0]
block13_sepconv2 (SeparableConv	(None, 8, 8, 1024)	752024	block13_sepconv2_act[0][0]
block13_sepconv2_bn (BatchNorma	(None, 8, 8, 1024)	4096	block13_sepconv2[0][0]
conv2d_51 (Conv2D)	(None, 4, 4, 1024)	745472	add_154[0][0]
block13_pool (MaxPooling2D)	(None, 4, 4, 1024)	0	block13_sepconv2_bn[0][0]
batch_normalization_51 (BatchNo	(None, 4, 4, 1024)	4096	conv2d_51[0][0]
add_155 (Add)	(None, 4, 4, 1024)	0	block13_pool[0][0] batch_normalization_51[0][0]
block14_sepconv1 (SeparableConv	(None, 4, 4, 1536)	1582080	add_155[0][0]
block14_sepconv1_bn (BatchNorma	(None, 4, 4, 1536)	6144	block14_sepconv1[0][0]
block14_sepconv1_act (Activatio	(None, 4, 4, 1536)	0	block14_sepconv1_bn[0][0]
block14_sepconv2 (SeparableConv	(None, 4, 4, 2048)	3159552	block14_sepconv1_act[0][0]
block14_sepconv2_bn (BatchNorma	(None, 4, 4, 2048)	8192	block14_sepconv2[0][0]
block14_sepconv2_act (Activatio	(None, 4, 4, 2048)	0	block14_sepconv2_bn[0][0]

==========

Total params: 20,861,480
Trainable params: 0
Non-trainable params: 20,861,480

(None, 128, 128, 3)

```
# Преобразуем изображение
inp = Input(shape=(None,None, 3))
out = Lambda(lambda image: ktf.image.resize(image, (128, 128)))(inp)

inputLayer = Model(inputs=inp, outputs=out, name="resizer")
inputLayer.summary()
```

```
Model: "resizer"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_12 (InputLayer)        [(None, None, None, 3)]   0
_________________________________________________________________
lambda_11 (Lambda)           (None, 128, 128, 3)       0
=================================================================
Total params: 0
Trainable params: 0
Non-trainable params: 0
_________________________________________________________________
```

```
# Создание модели составной сети
model = Sequential()
# Добавляем слой для преобразования размера изображения
model.add(inputLayer)
# Добавляем сверточные слои
model.add(xception)
# Преобразуем двумерный массив Xception в одномерный
model.add(Flatten())
# Полносвязный слой
model.add(Dense(256, activation='relu'))
# Слой регуляризации (для предотвращения переобучения)
model.add(Dropout(0.5))
# Кол-во классов
model.add(Dense(num_classes, activation='softmax'))

# Собираем модель
model.build(input_shape)
model.summary()
```

```
Model: "sequential_12"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
resizer (Functional)         (None, 128, 128, 3)       0
_________________________________________________________________
module_wrapper_12 (ModuleWra (None, 4, 4, 2048)        20861480
_________________________________________________________________
flatten_11 (Flatten)         (None, 32768)             0
_________________________________________________________________
dense_22 (Dense)             (None, 256)               8388864
_________________________________________________________________
dropout_11 (Dropout)         (None, 256)               0
_________________________________________________________________
dense_23 (Dense)             (None, 10)                2570
=================================================================
Total params: 29,252,914
Trainable params: 8,391,434
Non-trainable params: 20,861,480
_________________________________________________________________
```

```
[ ] # Компилируем составную сеть
    epochs = 5
    model.compile(loss='categorical_crossentropy',
                  optimizer=Adam(lr=1e-5),
                  metrics=['accuracy'])
```

```
[ ] model.fit(x_train, y_train, validation_data=(x_test, y_test), epochs = 6, batch_size=batch_size)
```

```
Epoch 1/6
1875/1875 [==============================] - 106s 55ms/step - loss: 1.6371 - accuracy: 0.4794 - val_loss: 0.7300 - val_accuracy: 0.8429
Epoch 2/6
1875/1875 [==============================] - 108s 58ms/step - loss: 0.7638 - accuracy: 0.7883 - val_loss: 0.5194 - val_accuracy: 0.8768
Epoch 3/6
1875/1875 [==============================] - 101s 54ms/step - loss: 0.6025 - accuracy: 0.8299 - val_loss: 0.4305 - val_accuracy: 0.8892
Epoch 4/6
1875/1875 [==============================] - 101s 54ms/step - loss: 0.5142 - accuracy: 0.8502 - val_loss: 0.3788 - val_accuracy: 0.9047
Epoch 5/6
1875/1875 [==============================] - 101s 54ms/step - loss: 0.4621 - accuracy: 0.8651 - val_loss: 0.3533 - val_accuracy: 0.9049
Epoch 6/6
1875/1875 [==============================] - 101s 54ms/step - loss: 0.4144 - accuracy: 0.8773 - val_loss: 0.3252 - val_accuracy: 0.9125
(60000, 28, 28, 3)
```

```
# Final evaluation of the model
scores = model.evaluate(x_test, y_test, verbose=0)
print("Accuracy: %.2f%%" % (scores[1]*100))
```

```
Accuracy: 91.25%
```