**Московский государственный технический университет им. Н.Э. Баумана**

Кафедра «Системы обработки информации и управления»

**Лабораторная работа №4**

**по курсу «Проектирование интеллектуальных систем»**

Выполнил:

Хотин П.Ю.

ИУ5-24М

Москва, 2020 год

Импорт библиотек:

import keras  
from keras.datasets import cifar10  
from keras.preprocessing.image import ImageDataGenerator  
from keras.models import Sequential  
from keras.layers import Dense, Dropout, Activation, Flatten  
from keras.layers import Conv2D, MaxPooling2D  
import os  
from keras.constraints import maxnorm  
from keras.optimizers import SGD

Using TensorFlow backend.

from keras.callbacks import ModelCheckpoint, TensorBoard  
import datetime

Задаем количество эпох и объем батча:

batch\_size = 32  
num\_classes = 10  
epochs = 5  
num\_predictions = 20

**Скачиваем датасет и приводим к виду для обучения с помощью нейронной сети:**

(x\_train, y\_train), (x\_test, y\_test) = cifar10.load\_data()  
print('x\_train shape:', x\_train.shape)  
print(x\_train.shape[0], 'train samples')  
print(x\_test.shape[0], 'test samples')  
x\_train = x\_train.astype('float32')  
x\_test = x\_test.astype('float32')  
x\_train = x\_train / 255.0  
x\_test = x\_test / 255.0  
y\_train = keras.utils.to\_categorical(y\_train, num\_classes)  
y\_test = keras.utils.to\_categorical(y\_test, num\_classes)

Видно, что набор состоит из 50000 тренировочных элементов и 10000 тестовых

x\_train shape: (50000, 32, 32, 3)  
50000 train samples  
10000 test samples

**Создание базовой модели:**

**def** create\_model():  
 model = Sequential()  
 model.add(Conv2D(32, (3, 3), input\_shape=(32, 32, 3), padding='same', activation='relu', kernel\_constraint=maxnorm(3)))  
 model.add(Dropout(0.2))  
 model.add(Conv2D(64, (3, 3), activation='relu', padding='same', kernel\_constraint=maxnorm(3)))  
 model.add(MaxPooling2D(pool\_size=(2, 2)))  
 model.add(Conv2D(128, (3, 3), input\_shape=(32, 32, 3), padding='same', activation='relu', kernel\_constraint=maxnorm(3)))  
 model.add(MaxPooling2D(pool\_size=(2, 2)))  
 model.add(Dropout(0.2))  
 model.add(Flatten())  
 model.add(Dense(512, activation='relu', kernel\_constraint=maxnorm(3)))  
 model.add(Dropout(0.5))  
 model.add(Dense(num\_classes, activation='softmax'))  
 lrate = 0.01  
 decay = lrate/epochs  
 sgd = SGD(lr=lrate, momentum=0.9, decay=decay, nesterov=False)  
 model.compile(loss='categorical\_crossentropy', optimizer=sgd, metrics=['accuracy'])  
 **return** model

model = create\_model()

Описание модели:

model.summary()

WARNING:tensorflow:From /Users/paulik/Универ/giis/env/lib/python3.7/site-packages/tensorflow\_core/python/ops/resource\_variable\_ops.py:1630: calling BaseResourceVariable.\_\_init\_\_ (from tensorflow.python.ops.resource\_variable\_ops) with constraint is deprecated and will be removed in a future version.  
Instructions for updating:  
If using Keras pass \*\_constraint arguments to layers.  
WARNING:tensorflow:From /Users/paulik/Универ/giis/env/lib/python3.7/site-packages/keras/backend/tensorflow\_backend.py:4070: The name tf.nn.max\_pool is deprecated. Please use tf.nn.max\_pool2d instead.  
  
Model: "sequential\_1"  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Layer (type) Output Shape Param #   
=================================================================  
conv2d\_1 (Conv2D) (None, 32, 32, 32) 896   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
dropout\_1 (Dropout) (None, 32, 32, 32) 0   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
conv2d\_2 (Conv2D) (None, 32, 32, 64) 18496   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
max\_pooling2d\_1 (MaxPooling2 (None, 16, 16, 64) 0   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
conv2d\_3 (Conv2D) (None, 16, 16, 128) 73856   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
max\_pooling2d\_2 (MaxPooling2 (None, 8, 8, 128) 0   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
dropout\_2 (Dropout) (None, 8, 8, 128) 0   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
flatten\_1 (Flatten) (None, 8192) 0   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
dense\_1 (Dense) (None, 512) 4194816   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
dropout\_3 (Dropout) (None, 512) 0   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
dense\_2 (Dense) (None, 10) 5130   
=================================================================  
Total params: 4,293,194  
Trainable params: 4,293,194  
Non-trainable params: 0  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Создание колбеков для чекпоинтов и тензорборда**

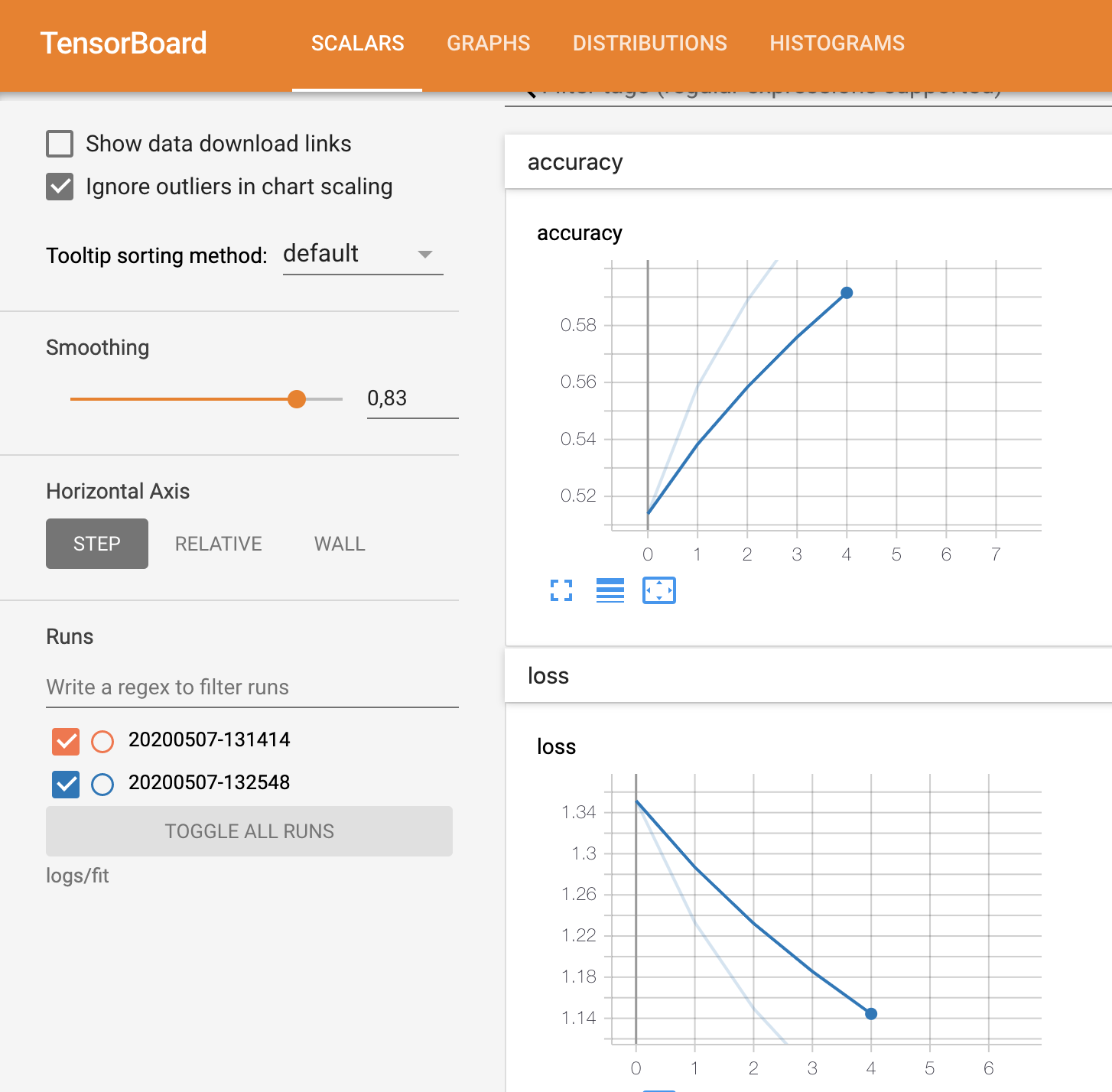
checkpoint\_path = "training\_chekpoints/cp.ckpt"  
checkpoint\_dir = os.path.dirname(checkpoint\_path)  
  
*# Создаем коллбек сохраняющий веса модели*  
cp\_callback = ModelCheckpoint(filepath=checkpoint\_path,  
 save\_weights\_only=True,  
 verbose=1, period=1)  
log\_dir = "logs/fit/" + datetime.datetime.now().strftime("%Y%m%d-%H%M%S")  
tensorboard\_callback = TensorBoard(log\_dir=log\_dir, histogram\_freq=1)

Начало тренировки модели:

model.fit(x\_train,  
 y\_train,  
 validation\_data=(x\_test, y\_test),  
 epochs=epochs,  
 batch\_size=batch\_size,  
 callbacks=[cp\_callback, tensorboard\_callback])

Train on 50000 samples, validate on 10000 samples  
Epoch 1/5  
50000/50000 [==============================] - 144s 3ms/step - loss: 1.3515 - accuracy: 0.5139 - val\_loss: 1.2078 - val\_accuracy: 0.5714  
  
Epoch 00001: saving model to training\_chekpoints/cp.ckpt  
WARNING:tensorflow:From /Users/paulik/Универ/giis/env/lib/python3.7/site-packages/keras/callbacks/tensorboard\_v1.py:343: The name tf.Summary is deprecated. Please use tf.compat.v1.Summary instead.  
  
Epoch 2/5  
50000/50000 [==============================] - 148s 3ms/step - loss: 1.2331 - accuracy: 0.5586 - val\_loss: 1.1382 - val\_accuracy: 0.5968  
  
Epoch 00002: saving model to training\_chekpoints/cp.ckpt  
Epoch 3/5  
50000/50000 [==============================] - 150s 3ms/step - loss: 1.1496 - accuracy: 0.5888 - val\_loss: 1.0749 - val\_accuracy: 0.6157  
  
Epoch 00003: saving model to training\_chekpoints/cp.ckpt  
Epoch 4/5  
50000/50000 [==============================] - 148s 3ms/step - loss: 1.0878 - accuracy: 0.6127 - val\_loss: 1.0217 - val\_accuracy: 0.6399  
  
Epoch 00004: saving model to training\_chekpoints/cp.ckpt  
Epoch 5/5  
50000/50000 [==============================] - 148s 3ms/step - loss: 1.0385 - accuracy: 0.6314 - val\_loss: 0.9857 - val\_accuracy: 0.6484  
  
Epoch 00005: saving model to training\_chekpoints/cp.ckpt  
<keras.callbacks.callbacks.History at 0x136d04b10>

Вывод метрик в Tensorboard:



**Сохранение полной модели**

*# os.getcwd()*  
save\_dir = os.path.join(os.getcwd(), 'models')  
print(save\_dir)  
model\_name = 'modelpkh.h5'  
model\_path = os.path.join(save\_dir, model\_name)  
model.save(model\_path)  
print('Saved trained model at %s ' % model\_path)

/Users/paulik/Универ/giis/models  
Saved trained model at /Users/paulik/Универ/giis/models/modelpkh.h5

*# Score trained model.*  
**def** model\_evaluate(model1):  
 scores = model1.evaluate(x\_test, y\_test, verbose=1)  
 print('Test loss:', scores[0])  
 print('Test accuracy:', scores[1])

**Проверка точности базовой модели без тренировки**

*#Ненатренированная модель*  
model2 = create\_model()  
model\_evaluate(model2)

WARNING:tensorflow:From /Users/paulik/Универ/giis/env/lib/python3.7/site-packages/keras/backend/tensorflow\_backend.py:422: The name tf.global\_variables is deprecated. Please use tf.compat.v1.global\_variables instead.  
  
10000/10000 [==============================] - 5s 520us/step  
Test loss: 2.306235139465332  
Test accuracy: 0.09300000220537186

*Точность показала 9%*

**Теперь создадим базовую модель и загрузим веса из чекпоинтов, полученных в результате обучения**

model\_trained = create\_model()  
model\_trained.load\_weights(checkpoint\_path)  
model\_evaluate(model\_trained)

10000/10000 [==============================] - 6s 579us/step  
Test loss: 0.9856774023056031  
Test accuracy: 0.6484000086784363

**Точность на тестовой выборке составила 64%**

**Ответы на вопросы:**

1) **Как включить TensorBoard?**

tensorboard --logdir logs/fit и далее перейти по ссылке

2) **Как сбросить граф?**

tf.keras.backend.clear\_session() для tfv.2.1

3) **Зачем нужны коллекции?**

Коллекция - это объект похожий на словарь, в котором мы храним элементы узлов графа.Например, в коллекцию сохраняется переменная измерения точности и входные элементы модели.

4) **Перечислите команды для добавления переменных в сводную статистику.**

Для записи дефолтных метрик:  
*log\_dir = "logs/fit/" + datetime.datetime.now().strftime("%Y%m%d-%H%M%S")*  
*tensorboard\_callback = TensorBoard(log\_dir=log\_dir, histogram\_freq=1)*  
Для записи дополнительных статистик в цикле итерации эпох используются команды:  
*with train\_summary\_writer.as\_default():*  
 *tf.summary.scalar('loss', train\_loss.result(), step=epoch)*  
 *tf.summary.scalar('accuracy', train\_accuracy.result(), step=epoch)*  
(для выборки обучения, в случае тестовой: *test\_summary\_writer.as\_default()*)

Список литературы

[1] Google. Tensorflow. 2018. Apr. url - https://www.tensorflow.org/api\_docs/python/tf/train/Saver.

[2] Google. TensorBoard. 2018. Apr. url - https://www.tensorflow.org/programmers\_guide/summaries\_and\_-

tensorboard.