Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»



**Лабораторная работа №4 по дисциплине**

«Проектирование интеллектуальных систем»

**ИСПОЛНИТЕЛЬ:**

Березин И. С.

Группа ИУ5-23М

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

"\_\_"\_\_\_\_\_\_\_\_\_\_\_ 2019 г.

Москва 2019

1. **Задание**

Модифицировать программный код лабораторной №3 с добавлением сохранения модели и сохранения сводных статистик для изучения Tensorboard. Написать дополнительный код, который покажет демонстрацию восстановления модели из файла с расширением ckpt.

1. **CIFAR10**

Модифицированный код программы:

def variable\_summaries(var):

with tf.name\_scope('summaries'):

mean = tf.reduce\_mean(var)

tf.summary.scalar('mean', mean)

with tf.name\_scope('stddev'):

stddev = tf.sqrt(tf.reduce\_mean(tf.square(var - mean)))

tf.summary.scalar('stddev', stddev)

tf.summary.scalar('max', tf.reduce\_max(var))

tf.summary.scalar('min', tf.reduce\_min(var))

tf.summary.histogram('histogram', var)

LOG\_DIR = "lab4/logs/summaries"

tf.reset\_default\_graph()

x = tf.placeholder(tf.float32, shape=[None, 3072]) # создаем переменную для x

y\_ = tf.placeholder(tf.float32, shape=[None, 10])

keep\_prob = tf.placeholder(tf.float32) # переменная для dropout

x\_image = tf.reshape(x, [-1, 32, 32, 3]) # преобразование в размеры картинки

with tf.name\_scope('conv\_1'):

conv1 = conv\_layer(x\_image, shape=[3, 3, 3, 32]) # создание сверточного слоя

conv1\_pool = max\_pool\_2x2(conv1) # слой пулинга

with tf.name\_scope('conv\_2'):

conv2 = conv\_layer(conv1\_pool, shape=[3, 3, 32, 64])

conv2\_pool = max\_pool\_2x2(conv2)

with tf.name\_scope('conv\_3'):

# conv3 = conv\_layer(conv2, shape=[3, 3, 64, 128])

conv3 = conv\_layer(conv2\_pool, shape=[3, 3, 64, 128])

conv3\_pool = max\_pool\_2x2(conv3)

conv3\_flat = tf.reshape(conv3\_pool, [-1, 4\*4\*128])

with tf.name\_scope('full\_1'):

full\_1 = tf.nn.relu(full\_layer(conv3\_flat, 512))

with tf.name\_scope('dropout'):

full1\_drop = tf.nn.dropout(full\_1, keep\_prob=keep\_prob)

with tf.name\_scope('activations'):

y\_conv = full\_layer(full1\_drop, 10)

variable\_summaries(y\_conv)

with tf.name\_scope('cross'):

cross\_entropy = tf.reduce\_mean(tf.nn.softmax\_cross\_entropy\_with\_logits\_v2(

logits=y\_conv,

labels=y\_))

tf.summary.scalar('cross\_entropy', cross\_entropy)

train\_step = tf.train.AdamOptimizer(learning\_rate=0.0005).minimize(cross\_entropy)

correct\_prediction = tf.equal(tf.argmax(y\_conv, 1), tf.argmax(y\_, 1))

with tf.name\_scope('accuracy'):

accuracy = tf.reduce\_mean(tf.cast(correct\_prediction , tf.float32))

tf.summary.scalar('accuracy', accuracy)

# Merge all the summaries

merged = tf.summary.merge\_all()

#saver

DIR = 'lab4/log'

train\_var = [x, y\_, accuracy, keep\_prob]

tf.add\_to\_collection('train\_var', train\_var[0])

tf.add\_to\_collection('train\_var', train\_var[1])

tf.add\_to\_collection('train\_var', train\_var[2])

tf.add\_to\_collection('train\_var', train\_var[3])

saver = tf.train.Saver(max\_to\_keep=7, keep\_checkpoint\_every\_n\_hours=0.5)

saver.export\_meta\_graph(os.path.join(DIR, "model\_ckpt.meta"),

collection\_list=['train\_var'])

with tf.Session() as sess:

train\_writer = tf.summary.FileWriter(LOG\_DIR + '/train',

graph=tf.get\_default\_graph())

test\_writer = tf.summary.FileWriter(LOG\_DIR + '/test',

graph=tf.get\_default\_graph())

sess.run(tf.global\_variables\_initializer())

start\_time = time.time()

for epochs in range(5):

for i in range(2500):

batch = [x\_train[20\*i:20\*(i+1)] , y\_train[20\*i:20\*(i+1)]]

if i % 250 == 0:

summary,train\_accuracy = sess.run([merged,accuracy],

feed\_dict={x: batch[0], y\_: batch[1], keep\_prob: 1.0})

print("time {}, step {}, epoch{} training accuracy {}"

.format(time.time() - start\_time, i, epochs, train\_accuracy))

# Add to summaries

train\_writer.add\_summary(summary, i+2500\*epochs)

sess.run(train\_step, feed\_dict={x: batch[0], y\_: batch[1],keep\_prob: 0.5})

saver.save(sess, 'my\_lab3')

X = x\_test.reshape(10, 1000, 3072)

Y = y\_test.reshape(10, 1000, 10)

test\_accuracy = []

for i in range(10):

summary, test\_accur = sess.run([merged,accuracy], feed\_dict={x:X[i], y\_:Y[i], keep\_prob:1.0})

test\_accuracy.append(test\_accur)

test\_writer.add\_summary(summary, i)

test\_accuracy = np.mean(test\_accuracy)

print("test accuracy: {}".format(test\_accuracy))

Результат программы:

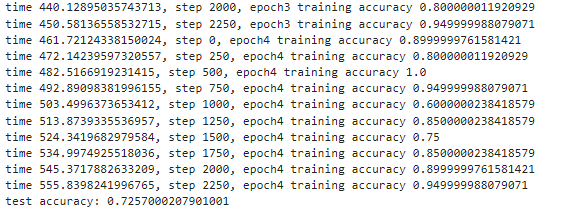


Рисунок 1. Результат выполнения программы распознавания набора данных CIFAR10

Код для восстановления модели:

import tensorflow as tf

from tensorflow.python.keras.datasets import cifar10

import numpy as np

import time

from keras.utils import np\_utils

from tensorflow.python import debug as tf\_debug

import os

tf.reset\_default\_graph()

(\_, \_), (X\_test, y\_test) = cifar10.load\_data()

nb\_classes = 10

y\_test = y\_test.reshape(y\_test.shape[0])

x\_test = X\_test.astype('float32')

x\_test /= 255

x\_test = np.reshape(x\_test, [-1, 3072])

y\_test = np\_utils.to\_categorical(y\_test, nb\_classes)

DIR1 = ''

DIR2 = 'lab4/log'

with tf.Session() as sess:

sess.run(tf.global\_variables\_initializer())

saver = tf.train.import\_meta\_graph(os.path.join(

DIR2,"model\_ckpt.meta"))

saver.restore(sess, os.path.join(DIR1,"my\_lab3"))

x = tf.get\_collection('train\_var')[0]

y\_ = tf.get\_collection('train\_var')[1]

accuracy = tf.get\_collection('train\_var')[2]

keep\_prob = tf.get\_collection('train\_var')[3]

X = x\_test.reshape(10, 1000, 3072)

Y = y\_test.reshape(10, 1000, 10)

test\_accuracy = np.mean([sess.run(accuracy, feed\_dict={x:X[i], y\_:Y[i], keep\_prob:1.0}) for i in range(10)])

print("test accuracy: {}".format(test\_accuracy))

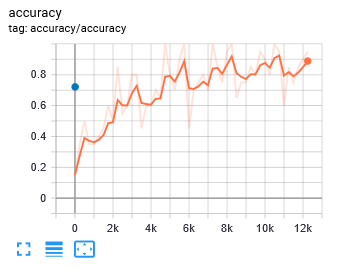


Рисунок 2. График изменения точности модели

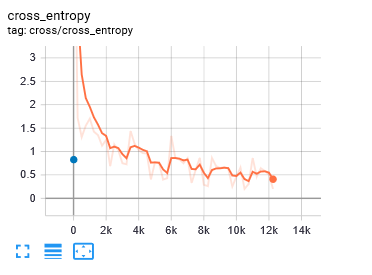


Рисунок 3. График изменения кросс-энтропии

1. **Список литературы**
2. Черненький И. М., Методические указания к лабораторной работе №3.
3. Николенко С.И., Кадурин А.А., Архангельская Е.О. Глубокое обучение. – Издательский дом "Питер", 2019. — 480 с.: ил. — (Серия «Библиотека программиста»).