Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»



**Лабораторная работа №2 по дисциплине**

«Проектирование интеллектуальных систем»

**ИСПОЛНИТЕЛЬ:**

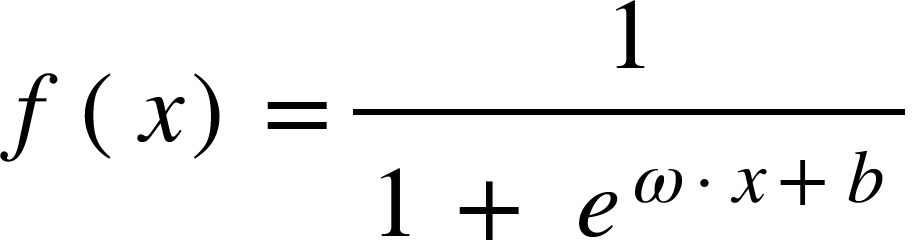
Чечнев А.А.

Группа ИУ5-23М

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

"\_\_"\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

Москва 2020

1. **Задание**
2. Создать логистическую регрессию для классификации набора данных MNIST. Функция логистической регрессии выглядит следующим образом: 
3. Создать нейронную сеть с 5 полносвязными слоями для классификации набора MNIST с количеством нейронов в слоях от первого до пятого (200, 100, 60, 30, 10)

**Логистическая регрессия**

%tensorflow\_version 1.x

import tensorflow as tf

tf.\_\_version\_\_

from tensorflow.examples.tutorials.mnist import input\_data

DATA\_DIR = 'tmp/data'

NUM\_STEPS = 1000

MINIBATCH\_SIZE = 100

LEARNING\_RATE = 0.5

**Создание вычислительного графа**

data = input\_data.read\_data\_sets(DATA\_DIR, one\_hot=True)

x = tf.placeholder (tf. float32 , [None, 784])

W = tf.Variable(tf.zeros([784, 10]))

b = tf.Variable(tf.zeros([10]), dtype =tf.float32)

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

#y\_pred = tf.matmul(x, W) + b

f = tf.matmul(x, W) + b

y\_pred = 1/(1+tf.exp(f))

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

                              logits=y\_pred, labels=y\_true))

gd\_step = tf.train.GradientDescentOptimizer(LEARNING\_RATE).minimize(cross\_entropy)

correct\_mask = tf.equal(tf.argmax(y\_pred, 1), tf.argmax(y\_true, 1))

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

**Запуск сессии**

with tf.Session() as sess:

    # Train

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

    for i in range ( NUM\_STEPS ):

        batch\_x , batch\_y = data.train.next\_batch(MINIBATCH\_SIZE)

        sess.run(gd\_step, feed\_dict ={ x : batch\_x , y\_true : batch\_y })

    ans = sess.run(accuracy, feed\_dict={x : data.test.images,

                                        y\_true:data.test.labels})

print ("Accuracy : {:.4}% ". format ( ans \*100))



**Нейронная сеть**

import tensorflow as tf

from tensorflow.examples.tutorials.mnist import input\_data

DATA\_DIR = 'tmp/data'

NUM\_STEPS = 1000

MINIBATCH\_SIZE = 100

LEARNING\_RATE = 0.5

# layers sizes

L1 = 200

L2 = 100

L3 = 60

L4 = 30

L5 = 10

data = input\_data.read\_data\_sets(DATA\_DIR, one\_hot = True)

x = tf.placeholder(tf.float32, [None, 784])

l1 = tf.layers.dense(x, L1, activation=tf.nn.relu, use\_bias=True)

l2 = tf.layers.dense(l1, L2, activation=tf.nn.relu, use\_bias=True )

l3 = tf.layers.dense(l2, L3, activation=tf.nn.relu, use\_bias=True )

l4 = tf.layers.dense(l3, L4, activation=tf.nn.relu, use\_bias=True )

y\_pred = tf.layers.dense(l4, L5, activation=tf.nn.relu, use\_bias=True)

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

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

                               logits=y\_pred, labels=y\_true ))

gd\_step = tf.train.GradientDescentOptimizer(LEARNING\_RATE).minimize(cross\_entropy)

correct\_mask = tf.equal(tf.argmax(y\_pred, 1), tf.argmax(y\_true, 1))

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

with tf. Session () as sess :

    # Train

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

    for i in range ( NUM\_STEPS ):

        batch\_x , batch\_y = data . train . next\_batch ( MINIBATCH\_SIZE )

        sess.run(gd\_step, feed\_dict={x : batch\_x, y\_true:batch\_y})

    ans = sess.run(accuracy, feed\_dict ={x:data.test.images,

                      y\_true:data.test.labels})

print (" Accuracy : {:.4}% ". format ( ans \*100))



# **Контрольные вопросы**

# Что такое Variable?

Узел графа c типом Variable имеет переменное значение. После прохода цикла обучения, новое значение узла Variable сохраняется. Его используют для обозначения весов модели.

# Что такое placeholder?

Узел типа placeholder обозначается в виде пустого значения, для дальнейшего вставления в него входных значений.

# Что такое функция потерь?

Функция потерь описывает правила расчета разницы истинных и предсказанных значений.

# Какие другие названия функции потери?

Функция стоимости, разница между предсказанием модели и истинным значением.

# Зачем нужна функция потери?

Для вычисления разницы между предсказанием модели и истинным значением входного вектора.

# Как запустить обучение модели?

# 

В TensorFlow сперва создается оптимизатор с помощью функции GradientDescentOptimizer() с желаемым значением шага обучения. Затем мы создаем операцию в вычислительном графе С помощью optimizer.minimize() и передаем в качестве аргумента функцию потери:

optimizer = tf. train . GradientDescentOptimizer ( learning\_rate )

train = optimizer . minimize ( loss )

И обучение начинается после того, как операция train передается в качестве аргумента методу sess.run().

# Что делает tf.global\_variables\_initializer()?

После запуска tf.global\_variables\_initializer () в сеансе переменные будут содержать значения, которые были указаны для хранения при их объявлении (tf.Variable) ...

# Что такое minibatch?

Срез по количеству входных примеров, для расчета на подвыборке, а не на всем множестве входных параметров. Используется в стохастическом градиентном спуске.

# 9. Какие бывают активационные функции

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»).