Московский государственный технический университет им. Н.Э. Баумана
Кафедра «Системы обработки информации и управления»

Лабораторная работа 2
по курсу
«Проектирование интеллектуальных систем»
Тема:
«Логистическая регрессия и полносвязная сеть»

Выполнил:
студент группы ИУ5-24М
Зубаиров В. А.

Проверил:

Москва — 2020 г.

1. Задание на лабораторную работу

1. Создать логистическую регрессию для классификации набора данных MNIST. Функция логистической регрессии выглядит следующим образом:
   ## $f(x) = \frac{1}{1+e^{\omega \cdot x + b}}$
2. Создать нейронную сеть с 5 полносвязными слоями для классификации набора данных MNIST с количеством нейронов в слоях от первого до пятого - (200,100,60,30,10)

1.1. 1. Логистическая регрессия

1.1.1. 1.1 Импорт библиотек

```
In [2]: import tensorflow as tf
        print("TensorFlow version is ", tf.__version__)
        from tensorflow.examples.tutorials.mnist import input_data
```

```
TensorFlow version is  1.15.0
```

1.1.2. 1.2 Placeholders для входных данных и истинных значений, Variables для весов и отступов

```
In [3]: 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])
```

```
In [6]: y_pred = 1 / (1 + tf.exp(tf.matmul(x, W) + b))
```

```
In [13]: DATA_DIR = 'tmp/data'
         NUM_STEPS = 1000
         MINIBATCH_SIZE = 100
         LEARNING_RATE = 0.5
         data = input_data.read_data_sets(DATA_DIR, one_hot=True)
```

```
Extracting tmp/data/train-images-idx3-ubyte.gz
Extracting tmp/data/train-labels-idx1-ubyte.gz
Extracting tmp/data/t10k-images-idx3-ubyte.gz
Extracting tmp/data/t10k-labels-idx1-ubyte.gz
```

```
In [14]: cross_entropy = tf. reduce_mean (tf. nn . softmax_cross_entropy_with_logi
         logits = y_pred , labels = y_true ))
         gd_step = tf. train . GradientDescentOptimizer ( LEARNING_RATE ). minimi
         correct_mask = tf. equal (tf. argmax ( y_pred , 1) , tf. argmax ( y_true
         accuracy = tf. reduce_mean (tf. cast ( correct_mask , tf. float32 ))
```

1.1.3. 1.3 Обучение модели

```
In [12]: 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 : dat

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

```
Accuracy : 88.59%
```

Точность при обучении вышла 88,59%

1.2. 2. Нейронная сеть

1.2.1. 2.1 Количейство нейронов в слоях

```
In [15]: # layers sizes
         L1 = 200
         L2 = 100
         L3 = 60
         L4 = 30
         L5 = 10
```

1.2.2. 2.2 Данные и слои

```
In [17]: 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])
```

```
Extracting tmp/data/train-images-idx3-ubyte.gz
Extracting tmp/data/train-labels-idx1-ubyte.gz
Extracting tmp/data/t10k-images-idx3-ubyte.gz
Extracting tmp/data/t10k-labels-idx1-ubyte.gz
```

1.2.3. 2.3 Кросс-энтропия

```
In [18]: cross_entropy = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(l
         gd_step = tf.train.GradientDescentOptimizer(LEARNING_RATE).minimize(cross
         correct_mask = tf.equal(tf.argmax(y_pred, 1), tf.argmax(y_true, 1))
         accuracy = tf.reduce_mean(tf.cast(correct_mask, tf.float32))
```

1.2.4. 2.4 Обучение

```
In [20]: with tf.Session() as sess:
             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

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

```
Accuracy : 86.44%
```

Точность обучения составила 86,44%

1.3. Список литературы

[1] Google. Tensorflow. 2018. Feb. url - https://www.tensorflow.org/install/install_windows.
[2] url - https://virtualenv.pypa.io/en/stable/userguide/.
[3] Microsoft. about_Execution_Policies. 2018. url - https://technet.microsoft.com/en-us/library/dd347641.aspx.
[4] Jupyter Project. Installing Jupyter. 2018. url - http://jupyter.org/install.