q-learning <https://colab.research.google.com/drive/1E2RViy7xmor0mhqskZV14_NUj2jMpJz3#scrollTo=fKun8LInsas9>

Пересмотреть (игра змея) <https://github.com/Platun0v/snake-gym>

Code <https://github.com/techwithtim/Number-Guesser-Neural-Net/blob/master/testNetwork.py>

install tensorflow <https://www.youtube.com/watch?v=VRbSKfzFkxo>

DSP <https://exponenta.ru/news/cifrovaya-obrabotka-signalov>

Machine learning <https://www.youtube.com/playlist?list=PLLssT5z_DsK-h9vYZkQkYNWcItqhlRJLN>

<https://www.youtube.com/playlist?list=PL3FW7Lu3i5JvHM8ljYj-zLfQRF3EO8sYv>

Neural Network https://github.com/selfedu-rus/neural-network

https://ourcodeworld.com/articles/read/1433/how-to-fix-tensorflow-warning-could-not-load-dynamic-library-cudart64-110dll-dlerror-cudart64-110dll-not-found

|  |
| --- |
| WEBCAM  import cv2 |
|  |  |
|  | face\_cascade\_db = cv2.CascadeClassifier(cv2.data.haarcascades + "haarcascade\_frontalface\_default.xml") |
|  |  |
|  | cap = cv2.VideoCapture(0) |
|  |  |
|  | while True: |
|  | success, img = cap.read() |
|  | #img = cv2.imread("IMG\_20191012\_145410\_3.jpg") |
|  | img\_gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY) |
|  |  |
|  | faces = face\_cascade\_db.detectMultiScale(img\_gray, 1.1, 19) |
|  | for (x,y,w,h) in faces: |
|  | cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0),2) |
|  |  |
|  | cv2.imshow('rez', img) |
|  | #cv2.waitKey() |
|  | if cv2.waitKey(1) & 0xff == ord('q'): |
|  | break |
|  |  |
|  | cap.release() |
|  | cv2.destroyAllWindows() |

From images

|  |
| --- |
| import cv2 |
|  |  |
|  | face\_cascade\_db = cv2.CascadeClassifier(cv2.data.haarcascades + "haarcascade\_frontalface\_default.xml") |
|  |  |
|  | img = cv2.imread("IMG\_20191012\_145410\_3.jpg") |
|  | img\_gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY) |
|  |  |
|  | faces = face\_cascade\_db.\ |
|  | detectMultiScale(img, 1.1, 19) |
|  | for (x,y,w,h) in faces: |
|  | cv2.rectangle(img, (x,y), |
|  | (x+w,y+h), (0,255,0),2) |
|  |  |
|  | cv2.imshow('rez', img) |
|  | cv2.waitKey() |

EYES

|  |
| --- |
| import cv2 |
|  |  |
|  | face\_cascade\_db = cv2.CascadeClassifier(cv2.data.haarcascades+"haarcascade\_frontalface\_default.xml") |
|  | eye\_cascade = cv2.CascadeClassifier(cv2.data.haarcascades+"haarcascade\_eye.xml") |
|  |  |
|  | cap = cv2.VideoCapture(0) |
|  |  |
|  | while True: |
|  | success, img = cap.read() |
|  | #img = cv2.imread("IMG\_20191012\_145410\_3.jpg") |
|  | img\_gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY) |
|  | faces = face\_cascade\_db.detectMultiScale(img\_gray, 1.1, 19) |
|  | for (x,y,w,h) in faces: |
|  | cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 2) |
|  | img\_gray\_face = img\_gray[y:y+h,x:x+w] |
|  | eyes = eye\_cascade.detectMultiScale(img\_gray\_face, 1.1, 19) |
|  | for (ex, ey, ew, eh) in eyes: |
|  | cv2.rectangle(img, (x+ex, y+ey), (x+ex + ew, y+ey + eh), (255, 0, 0), 2) |
|  | cv2.imshow('rez', img) |
|  | if cv2.waitKey(1) & 0xff == ord('q'): |
|  | break |
|  | cap.release() |
|  | cv2.destroyAllWindows() |

Doesn’t work

import tensorflow as tf  
import numpy as np  
import os  
os.environ['TF\_CPP\_MIN\_LOG\_LEVEL']='3'  
model=tf.keras.models.Sequential()  
model.add(tf.keras.layers.Dense(1, input\_dim=1))  
model.compile(loss='mean\_squared\_error', optimizer='sgd')  
xs = np.array([-1, 0, 1, 2, 3, 4])  
ys = np.array([-3, -1, 1, 3, 5, 7])  
model.flit(xs, ys, epochs=500)  
  
to\_predict = np.array([10, 11, 12, 13])  
print(model.predict(to\_predict))

Базовая классификация (не работает)

# TensorFlow and tf.keras  
import tensorflow as tf  
  
# Helper libraries  
import numpy as np  
import matplotlib.pyplot as plt  
import os  
import zipfile  
local\_zip = 'train-images-idx3-ubyte.gz'  
zip\_ref = zipfile.ZipFile(local\_zip, 'r')  
zip\_ref.extectall('/tmp/')  
zip\_ref.close()  
  
fashion\_mnist = tf.keras.datasets.fashion\_mnist  
  
with open ('train-images-idx3-ubyte.gz', 'rb') as f:  
 train\_images = extract\_images(f)  
  
(train\_images, train\_labels), (test\_images, test\_labels) = fashion\_mnist.load\_data()  
class\_names = ['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat',  
 'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Ankle boot']  
train\_images.shape

import keras  
import numpy as np  
import tensorflow as tf  
import matplotlib.pyplot as plt  
  
# FOR TESTING  
  
mnist = tf.keras.datasets.mnist  
(x\_train, y\_train),(x\_test, y\_test) = mnist.load\_data()  
x\_test = tf.keras.utils.normalize(x\_test, axis=1)  
for test in range(len(x\_test)):  
 for row in range(28):  
 for x in range(28):  
 if x\_test[test][row][x] != 0:  
 x\_test[test][row][x] = 1  
  
  
model = tf.keras.models.load\_model('m.model')  
print(len(x\_test))  
predictions = model.predict(x\_test[:10])  
  
count = 0  
for x in range(len(predictions)):  
 guess = (np.argmax(predictions[x]))  
 actual = y\_test[x]  
 print("I predict this number is a:", guess)  
 print("Number Actually Is a:", actual)  
 if guess != actual:  
 #print("--------------")  
 #print('WRONG')  
 #print('---------------')  
 count+=1  
 plt.imshow(x\_test[x], cmap=plt.cm.binary)  
 plt.show()  
  
print("The program got", count, 'wrong, out of', len(x\_test))  
print(str(100 - ((count/len(x\_test))\*100)) + '% correct')

h=tf.matmul(W,X)  
  
hypothesis=tf.div(1., 1.+tf.exp(-h))  
cost=tf.reduce\_mean(-t\*tf.log(hypothesis)-(1-t)\*tf.log(1-hypothesis))  
a=tf.Variable(0,1)  
  
optimizer=tf.train.GradientDescentoptimizer(a)  
  
train=optimizer.minimize(cost)  
for step in xrange(2001):  
 with tf.compat.v1.Session() as sess:  
print (sess.run(train.feed\_dict={X:x\_data, Y:y\_data}))