**Жуковский Павел, 3 курс, 12 группа, кафедра КТС**

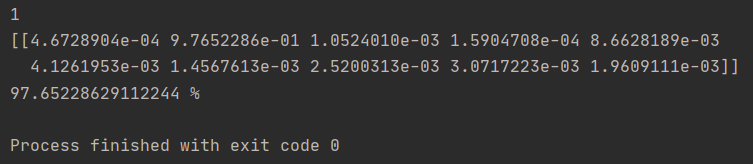
**Задание 1.**

Используйте обученные сверточные нейронные сети для распознавания своих объектов и рукописных цифр.Pl

Код для распознавания цифр (cifars\_rec.py):

import numpy as np  
from keras.preprocessing import image  
from tensorflow.keras.models import model\_from\_json  
  
# Загружаем изображение  
img\_path = '1.png'  
img = image.load\_img(img\_path, target\_size=(28, 28), color\_mode="grayscale")  
  
# Преобразуем изображение в массив numpy  
x = image.img\_to\_array(img)  
  
# Инвертируем и нормализуем изображение  
x = 255 - x  
x /= 255  
x = np.expand\_dims(x, axis=0)  
json\_file = open("mnist.json", "r")  
loaded\_model\_json = json\_file.read()  
json\_file.close()  
loaded\_model = model\_from\_json(loaded\_model\_json)  
loaded\_model.load\_weights("mnist.h5")  
loaded\_model.compile(loss="categorical\_crossentropy", optimizer="adam", metrics=["accuracy"])  
prediction = loaded\_model.predict(x)  
print(np.argmax(prediction))  
  
print(prediction)  
print(prediction[0][np.argmax(prediction)]\*100, '%')

Здесь мы загружаем уже обученную сеть из json файла mnist.json и затем используем для распознавания **цифр**. Вывод при выполнении:



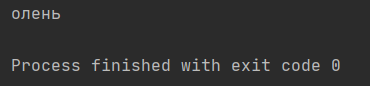
Цифру «1» программа распознала примерно на 97.7%.

Код для распознавания картинок (pictures\_rec.py):

import numpy as np  
from keras.preprocessing import image  
from tensorflow.keras.models import model\_from\_json  
  
img\_path = 'deer.png'  
img = image.load\_img(img\_path, target\_size=(32, 32))  
x = image.img\_to\_array(img)  
x /= 255  
x = np.expand\_dims(x, axis=0)  
json\_file = open("cifar\_model.json", "r")  
loaded\_model\_json = json\_file.read()  
json\_file.close()  
loaded\_model = model\_from\_json(loaded\_model\_json)  
loaded\_model.load\_weights("cifar\_model.h5")  
loaded\_model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])  
prediction = loaded\_model.predict(x)  
classes=['самолет', 'автомобиль', 'птица', 'кот', 'олень', 'собака', 'лягушка', 'лошадь', 'корабль', 'грузовик']  
print(classes[np.argmax(prediction)])

Здесь уже мы загружаем обученную сеть из json файла, и затем используем для распознавания **картинок**.

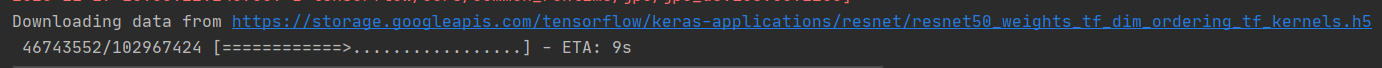
Изображение оленя было подано на вход и в результате программа определила, что это действительно олень. Вывод:



**Задание 2.**

Используйте предварительно обученную сеть VGG16 в Keras для распознавания своего изображения.

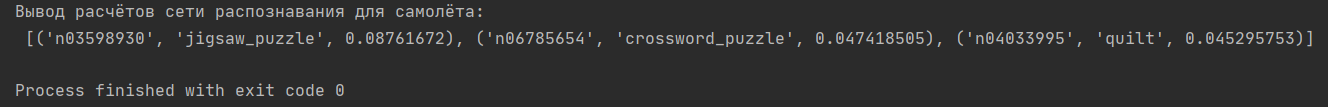
Вот таким образом моя среда разработки устанавливала сеть VGG16:



Здесь я поэкспериментировал с изображением самолёта и подавал его на вход этой сети. Код:

from keras.applications.vgg16 import VGG16  
from keras.preprocessing import image  
from keras.applications.vgg16 import preprocess\_input, decode\_predictions  
import numpy as np  
  
model = VGG16(weights='imagenet')  
img = image.load\_img('plane.jpg', target\_size=(224, 224))  
x = image.img\_to\_array(img)  
x = np.expand\_dims(x, axis=0)  
x = preprocess\_input(x)  
preds = model.predict(x)  
print('Вывод расчётов сети распознавания для самолёта:\n', decode\_predictions(preds, top=3)[0])

Результаты работы программы:



Насколько я понял, эта сеть сравнивала изображение самолёта с изображениями пазлов, кроссвордов и лоскутного одеяла и потому определила, что самолёт почти не похож ни на один из этих предметов. Мы можем видеть это по цифрам рядом с названиями предметов: 0.08761672, 0.047418505, 0.045295753.

**Задание 3.**

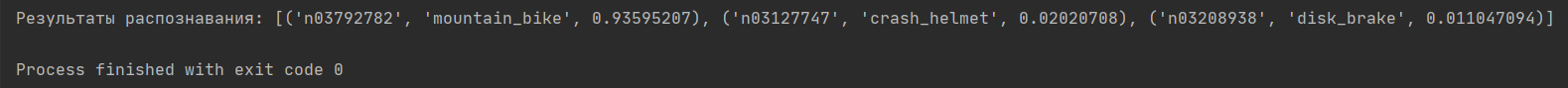
Используйте предварительно обученную сеть ResNet50 в Keras для распознавания своего изображения.

Установка сети ResNet50 произошла по аналогии с установкой сети VGG16 из предыдущего задания.

Для данной сети я использовал изображение велосипеда. Код:

from keras.applications.resnet50 import ResNet50  
from keras.preprocessing import image  
from keras.applications.resnet50 import preprocess\_input, decode\_predictions  
import numpy as np  
  
model = ResNet50(weights='imagenet')  
img = image.load\_img('велосипед.jpg', target\_size=(224, 224))  
x = image.img\_to\_array(img)  
x = np.expand\_dims(x, axis=0)  
x = preprocess\_input(x)  
preds = model.predict(x)  
print('Результаты распознавания:', decode\_predictions(preds, top=3)[0])

Вывод:



Интересно отметить, что сеть показала, что изображение велосипеда примерно на 93,6% похоже на изображение горного велосипеда, что в общем-то логично.