CODE DOCUMENT

SEGMENTATION OF BUTTERFLIES IMAGES IN OUTDOOR AREAS

CODE:

```
import numpy as np

import matplotlib.pyplot as plt

import os

import cv2


DATADIR = r"C:\Users\Dell\Desktop\CV project"

CATEGORIES = ['maniola_jurtina', 'monarch']


for category in CATEGORIES:

    path = os.path.join(DATADIR, category) # path to directory for images

    for img in os.listdir(path):

        img_array = cv2.imread(os.path.join(path,img), cv2.IMREAD_GRAYSCALE)

        plt.imshow(img_array, cmap="gray")

        break

    break



IMG_SIZE = 70


new_array = cv2.resize(img_array, (IMG_SIZE, IMG_SIZE))

plt.imshow(new_array, cmap = 'gray')

plt.show()


training_data = []


def create_training_data():
    for category in CATEGORIES:
        path = os.path.join(DATADIR, category) # path to directory for images
        class_num = CATEGORIES.index(category)
        for img in os.listdir(path):
            try:
                img_array = cv2.imread(os.path.join(path,img), cv2.IMREAD_GRAYSCALE)
                new_array = cv2.resize(img_array, (IMG_SIZE, IMG_SIZE))
                training_data.append([new_array, class_num])
            except Exception as e:
                pass

create_training_data()
print(len(training_data))
import random
random.shuffle(training_data)
import numpy as np

x = []
y = []
for features, label in training_data:
    x.append(features)
    y.append(label)

x = np.array(x).reshape(-1, IMG_SIZE, IMG_SIZE, 1)
import pickle

pickle_out = open("x.pickle", "wb")
pickle.dump(x, pickle_out)
pickle_out.close()

pickle_out = open("y.pickle", "wb")
```

```
pickle.dump(y, pickle_out)
pickle_out.close()
pickle_in = open("x.pickle", "rb")
x = pickle.load(pickle_in)



import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Activation, Flatten, Conv2D, MaxPooling2D
from tensorflow.keras.callbacks import TensorBoard
import pickle
import time

NAME = "maniola-vs-monarch-cnn-64x2-{}".format(int(time.time()))

tensorboard = TensorBoard(log_dir='logs/{}'.format(NAME))



x = np.array(pickle.load(open("x.pickle", "rb")))
y = np.array(pickle.load(open("y.pickle", "rb")))

x = x/255.0

model = Sequential()

model.add(Conv2D(64, (3,3), input_shape = x.shape[1:]))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(64, (3,3)))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2,2)))


model.add(Flatten())


model.add(Dense(64))
model.add(Activation("relu"))


model.add(Dense(1))
model.add(Activation('sigmoid'))


model.compile(loss='binary_crossentropy',
            optimizer='adam',
            metrics=['accuracy'])


model.fit(x, y, batch_size=32,epochs = 10, validation_split=0.1, callbacks=[tensorboard])
model.save('64x2-CNN.model')
```

Testing:

```
import cv2
import tensorflow as tf


CATEGORIES = ["maniola_jurtina", "monarch"]




def prepare(filepath):
    IMG_SIZE = 70
    img_array = cv2.imread(filepath, cv2.IMREAD_GRAYSCALE)
    new_array = cv2.resize(img_array, (IMG_SIZE,IMG_SIZE))
    return new_array.reshape(-1, IMG_SIZE, IMG_SIZE,1)
```

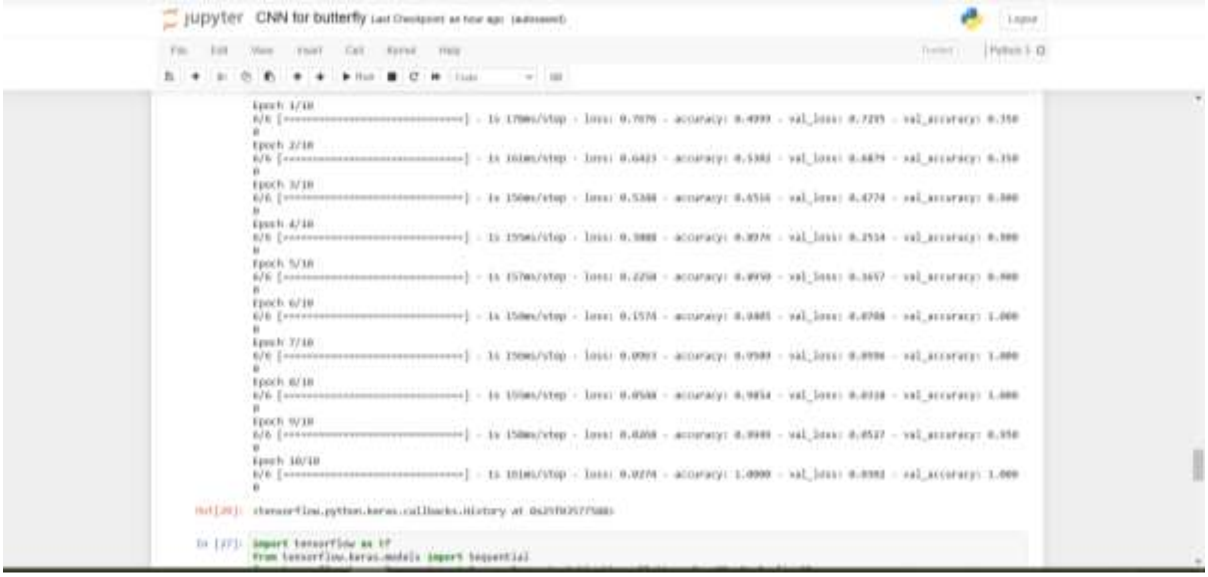
model = tf.keras.models.load_model("64x2-CNN.model")

prediction = model.predict([prepare(r'C:\Users\Dell\Pictures\Saved Pictures\smth.jpg')])

print(CATEGORIES[int(prediction[0][0])])

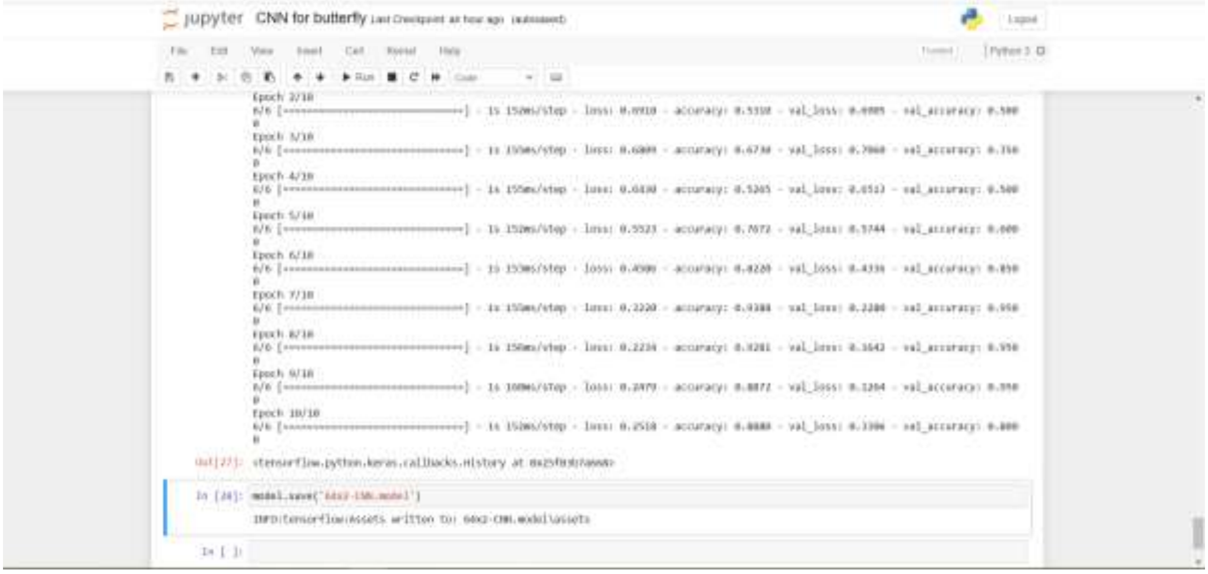
SCREENSHOTS:

Training:


jupyter CNN for butterfly Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Code

```
Epoch 1/10
6/6 [==============================] - 1s 170ms/step - loss: 0.7076 - accuracy: 0.4999 - val_loss: 0.7295 - val_accuracy: 0.350
0
Epoch 2/10
6/6 [==============================] - 1s 161ms/step - loss: 0.6423 - accuracy: 0.5382 - val_loss: 0.6879 - val_accuracy: 0.350
0
Epoch 3/10
6/6 [==============================] - 1s 150ms/step - loss: 0.5348 - accuracy: 0.6516 - val_loss: 0.4774 - val_accuracy: 0.800
0
Epoch 4/10
6/6 [==============================] - 1s 155ms/step - loss: 0.3888 - accuracy: 0.8976 - val_loss: 0.2514 - val_accuracy: 0.900
0
Epoch 5/10
6/6 [==============================] - 1s 157ms/step - loss: 0.2250 - accuracy: 0.8950 - val_loss: 0.3657 - val_accuracy: 0.900
0
Epoch 6/10
6/6 [==============================] - 1s 158ms/step - loss: 0.1574 - accuracy: 0.9485 - val_loss: 0.0708 - val_accuracy: 1.000
0
Epoch 7/10
6/6 [==============================] - 1s 150ms/step - loss: 0.0983 - accuracy: 0.9580 - val_loss: 0.0896 - val_accuracy: 1.000
0
Epoch 8/10
6/6 [==============================] - 1s 155ms/step - loss: 0.0588 - accuracy: 0.9854 - val_loss: 0.0318 - val_accuracy: 1.000
0
Epoch 9/10
6/6 [==============================] - 1s 158ms/step - loss: 0.0458 - accuracy: 0.9949 - val_loss: 0.0527 - val_accuracy: 0.950
0
Epoch 10/10
6/6 [==============================] - 1s 161ms/step - loss: 0.0274 - accuracy: 1.0000 - val_loss: 0.0392 - val_accuracy: 1.000
0
Out[26]: <tensorflow.python.keras.callbacks.History at 0x25f0257758>

In [27]: import tensorflow as tf
         from tensorflow.keras.models import Sequential
```

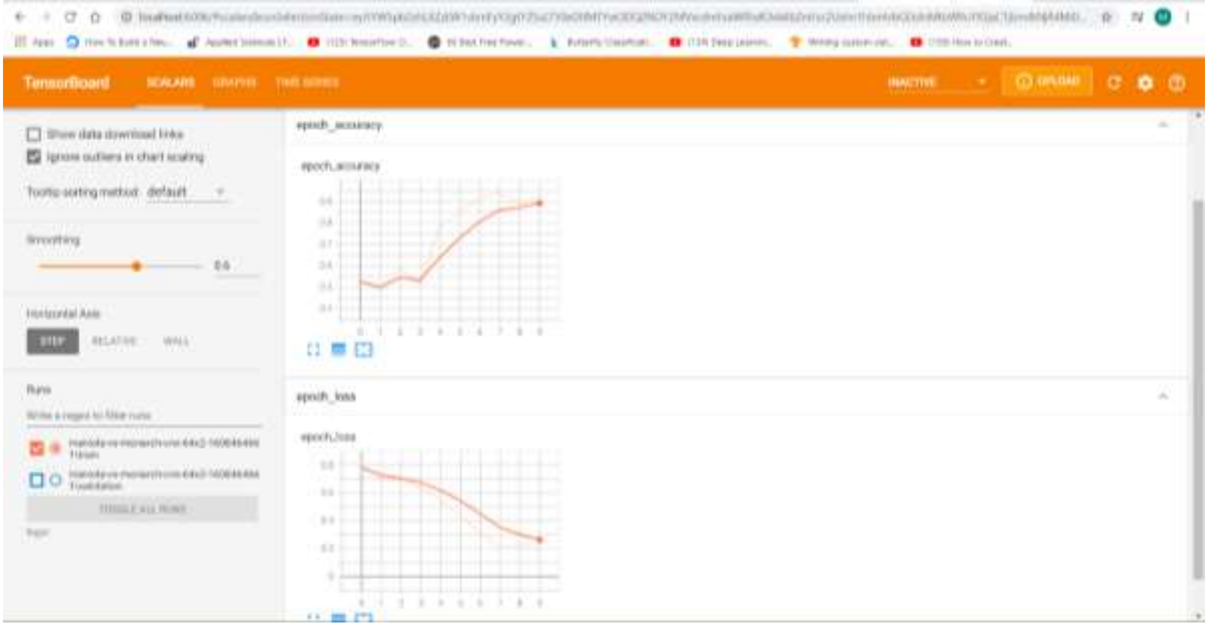
Saving CNN model



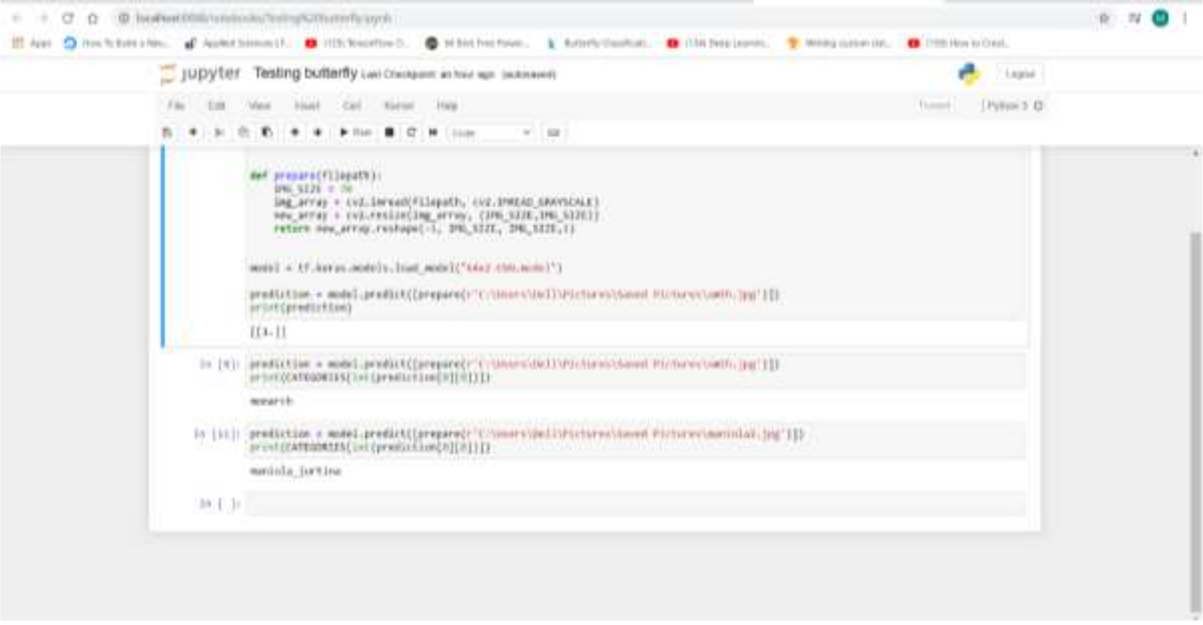
jupyter CNN for butterfly Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Code

```
Epoch 2/10
6/6 [==============================] - 1s 152ms/step - loss: 0.6910 - accuracy: 0.5310 - val_loss: 0.6905 - val_accuracy: 0.500
0
Epoch 3/10
6/6 [==============================] - 1s 155ms/step - loss: 0.6809 - accuracy: 0.6730 - val_loss: 0.7060 - val_accuracy: 0.350
0
Epoch 4/10
6/6 [==============================] - 1s 155ms/step - loss: 0.6430 - accuracy: 0.5265 - val_loss: 0.6513 - val_accuracy: 0.500
0
Epoch 5/10
6/6 [==============================] - 1s 152ms/step - loss: 0.5523 - accuracy: 0.7672 - val_loss: 0.5744 - val_accuracy: 0.600
0
Epoch 6/10
6/6 [==============================] - 1s 153ms/step - loss: 0.4500 - accuracy: 0.8220 - val_loss: 0.4336 - val_accuracy: 0.850
0
Epoch 7/10
6/6 [==============================] - 1s 155ms/step - loss: 0.2220 - accuracy: 0.9388 - val_loss: 0.2280 - val_accuracy: 0.950
0
Epoch 8/10
6/6 [==============================] - 1s 150ms/step - loss: 0.2234 - accuracy: 0.9281 - val_loss: 0.1642 - val_accuracy: 0.950
0
Epoch 9/10
6/6 [==============================] - 1s 160ms/step - loss: 0.2479 - accuracy: 0.8872 - val_loss: 0.1264 - val_accuracy: 0.950
0
Epoch 10/10
6/6 [==============================] - 1s 152ms/step - loss: 0.2518 - accuracy: 0.8888 - val_loss: 0.1306 - val_accuracy: 0.800
0
Out[27]: <tensorflow.python.keras.callbacks.History at 0x25f03b7a0a0>

In [28]: model.save('64x2-CNN.model')
         INFO:tensorflow:Assets written to: 64x2-CNN.model\assets

In [ ]:
```

TensorBoard accuracy graph



Testing:



```
def prepare(filepath):
    IMG_SIZE = 70
    img_array = cv2.imread(filepath, cv2.IMREAD_GRAYSCALE)
    new_array = cv2.resize(img_array, (IMG_SIZE,IMG_SIZE))
    return new_array.reshape(-1, IMG_SIZE, IMG_SIZE,1)


model = tf.keras.models.load_model("64x2-CNN.model")

prediction = model.predict([prepare(r'C:\Users\Dell\Pictures\Saved Pictures\smth.jpg')])
print(prediction)

[[1.]]
```

```
In [8]: prediction = model.predict([prepare(r'C:\Users\Dell\Pictures\Saved Pictures\smth.jpg')])
        print(CATEGORIES[int(prediction[0][0])])

        monarch
```

```
In [11]: prediction = model.predict([prepare(r'C:\Users\Dell\Pictures\Saved Pictures\maniolat.jpg')])
         print(CATEGORIES[int(prediction[0][0])])

         maniola_jurtina
```

```
In [ ]:
```

THE END