**Лабораторная работа на тему: определение эмоции человека по фото при помощи сверточной нейронной сети (CNN)**

1. Первым шагом необходимо определиться с набором данных для обучения (далее датасет) нейронной сети. В качестве датасета был выбран набор данных с Kaggle.

Данные представляют каталог, рисунок 1, который содержит в себе 5 подкаталогов: с изображениями лиц людей изображающих различные эмоции.

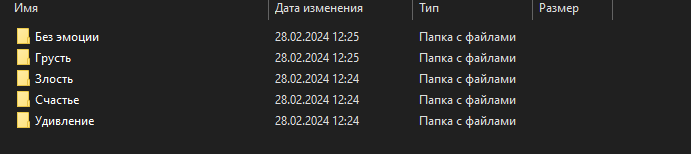


Рисунок 1 – Каталог изображений

Пример изображений из каталога Счастье, рисунок 2.



Рисунок 2 - Счастье

Вторым шагом пишем код нейронной сети и определяемся с её архитектурой и методом считывания данных из набора. Загружаем датасет указывая путь, указываем расширения файлов, определяем количество слоев, определяем количество фильтров на каждом сверточном слое, количество пикселей для свертки.

Dataset = imageDatastore('Dataset', 'IncludeSubfolders', true, 'LabelSource', 'foldernames');

[Training\_Data, Validation\_Data] = splitEachLabel(Dataset, 0.7,'randomized');

Resized\_Training\_Data = augmentedImageDatastore([256 256], Training\_Data);

Resized\_Validation\_Data = augmentedImageDatastore([256 256], Validation\_Data);

layers = [

imageInputLayer([256 256 3])

convolution2dLayer(5,8,Padding="same")

batchNormalizationLayer

reluLayer

maxPooling2dLayer(2,Stride=2)

convolution2dLayer(5,16,Padding="same")

batchNormalizationLayer

reluLayer

maxPooling2dLayer(2,Stride=2)

convolution2dLayer(5,32,Padding="same")

batchNormalizationLayer

reluLayer

maxPooling2dLayer(2,Stride=2)

convolution2dLayer(5,64,Padding="same")

batchNormalizationLayer

reluLayer

maxPooling2dLayer(2,Stride=2)

convolution2dLayer(5,64,Padding="same")

batchNormalizationLayer

reluLayer

maxPooling2dLayer(2,Stride=2)

convolution2dLayer(5,32,Padding="same")

batchNormalizationLayer

reluLayer

maxPooling2dLayer(2,Stride=2)

convolution2dLayer(5,16,Padding="same")

batchNormalizationLayer

reluLayer

maxPooling2dLayer(2,Stride=2)

convolution2dLayer(5,8,Padding="same")

batchNormalizationLayer

reluLayer

fullyConnectedLayer(5)

softmaxLayer

classificationLayer];

Training\_Options = trainingOptions('sgdm', ...

'MiniBatchSize', 4, ...

'MaxEpochs', 20, ...

'InitialLearnRate', 4e-4, ...

'Shuffle', 'every-epoch', ...

'ValidationData', Resized\_Validation\_Data, ...

'ValidationFrequency', 3, ...

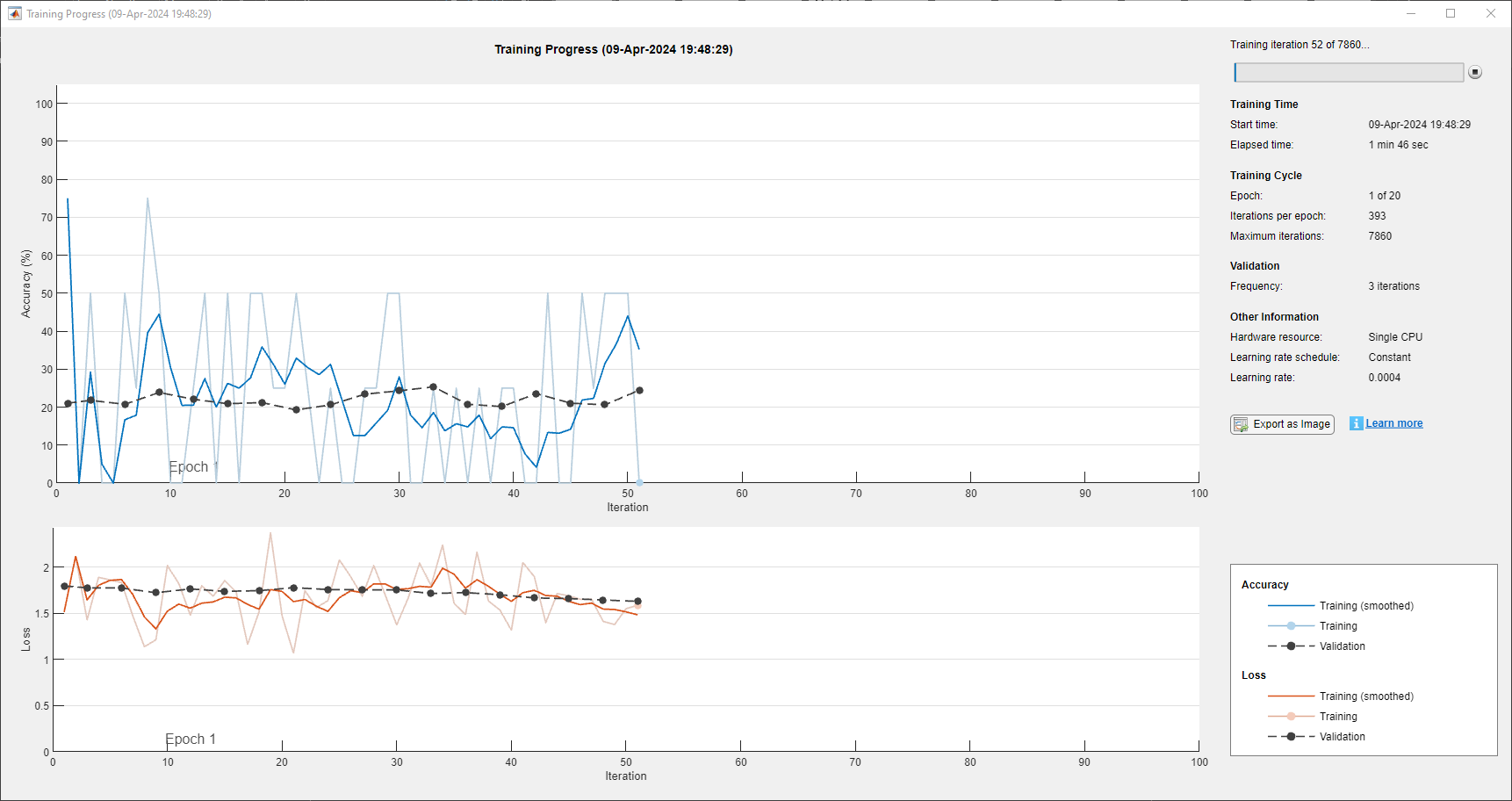
'Verbose', false, ...

'Plots', 'training-progress');

net = trainNetwork(Resized\_Training\_Data, layers, Training\_Options);

save('Face\_Recognizer12.mat', 'net');

1. Запускаем обучение сети



Следим за параметрами обучения, если они низкие, то нужно будет корректировать код создания сети, количество слоев и т.д.

1. Далее нужно создать код для тестирования сети test\_network.m на тестовых данных после обучения и сохранения модели, в качестве тестовых данных мы используем данные, которые не участвовали в обучении модели и наблюдаем за откликом сети.

Dataset = imageDatastore('Dataset', 'IncludeSubfolders', true, 'LabelSource', 'foldernames');

[Training\_Data, Validation\_Data] = splitEachLabel(Dataset, 0.7,'randomized');

Resized\_Training\_Data = augmentedImageDatastore([256 256], Training\_Data);

Resized\_Validation\_Data = augmentedImageDatastore([256 256], Validation\_Data);

loaded\_Network = load('Face\_Recognizer12.mat');

net = loaded\_Network.net;

[Label, Probability] = classify(net, Resized\_Validation\_Data);

accuracy = mean(Label == Validation\_Data.Labels)

index = randperm(numel(Validation\_Data.Files), 12);

figure

for i = 1:12

subplot(3,4,i)

I = readimage(Validation\_Data, index(i));

imshow(I)

actual = Validation\_Data.Labels(index(i))

label = Label(index(i));

title(string(label) + ", " + num2str(100\*max(Probability(index(i), :)), 3) + "% " + string(actual));

end

1. Тестируем модель.

