

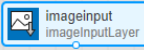
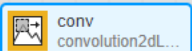
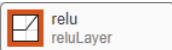
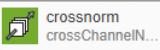
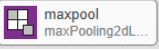
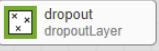

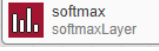
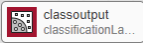
Отчет о выполнении индивидуального задания по теме
«Введение в сверточные нейронные сети»

Студента(-ки) группы 09-813

Махмутов Ринат

Задание

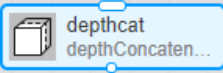
Создать простейшую нейронную сеть в , используя входной слой

, слои свертки  (1-2 слоя), relu слой , слой нормировки (например, ) , слой pooling (например, ) , слой dropout , полносвязный слой для классификации , и слой softmax  и собственно слой классификации .

Создать несколько сетей:

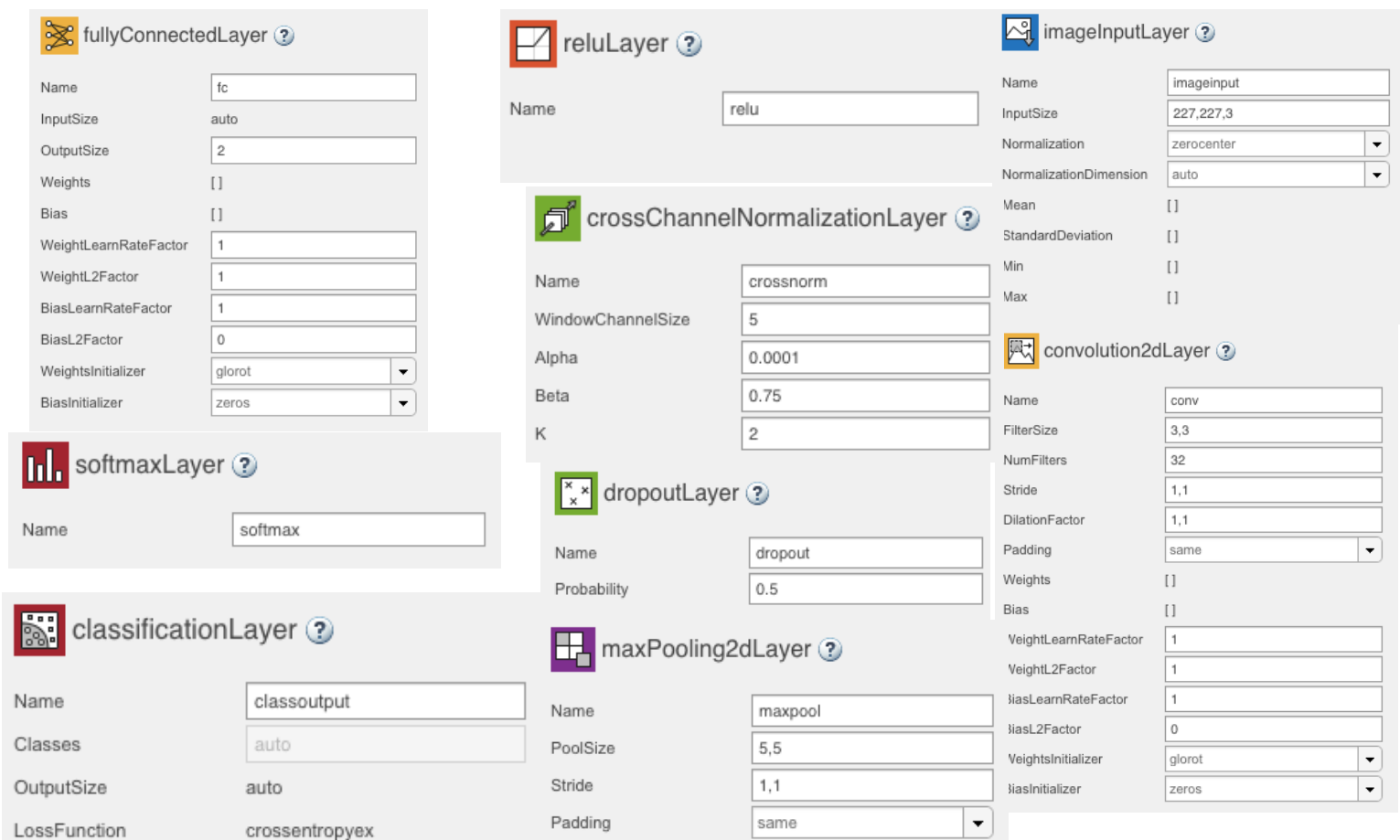
1) Последовательную сеть с параметрами по умолчанию в matlab, изменив только в полносвязном слое число классов для классификации: по умолчанию их 10, а для наших данных – 2 класса.

2) Разветвленную сеть с несколькими сверточными слоями с различными

размерами фильтров, используя слой .

Привести оценки качества классификации для каждой сети. Провести сравнение с другими методами классификации.

1) На рисунке ниже красным прямоугольником выделен пример требуемой архитектуры сети. Рядом приведены параметры слоев по умолчанию.



fullyConnectedLayer

Name: fc

InputSize: auto

OutputSize: 2

Weights: []

Bias: []

WeightLearnRateFactor: 1

WeightL2Factor: 1

BiasLearnRateFactor: 1

BiasL2Factor: 0

WeightsInitializer: glorot

BiasInitializer: zeros

reluLayer

Name: relu

crossChannelNormalizationLayer

Name: crossnorm

WindowChannelSize: 5

Alpha: 0.0001

Beta: 0.75

K: 2

dropoutLayer

Name: dropout

Probability: 0.5

maxPooling2dLayer

Name: maxpool

PoolSize: 5,5

Stride: 1,1

Padding: same

imageInputLayer

Name: imageinput

InputSize: 227,227,3

Normalization: zerocenter

NormalizationDimension: auto

Mean: []

StandardDeviation: []

Min: []

Max: []

convolution2dLayer

Name: conv

FilterSize: 3,3

NumFilters: 32

Stride: 1,1

DilationFactor: 1,1

Padding: same

Weights: []

Bias: []

WeightLearnRateFactor: 1

WeightL2Factor: 1

BiasLearnRateFactor: 1

BiasL2Factor: 0

WeightsInitializer: glorot

BiasInitializer: zeros

softmaxLayer

Name: softmax

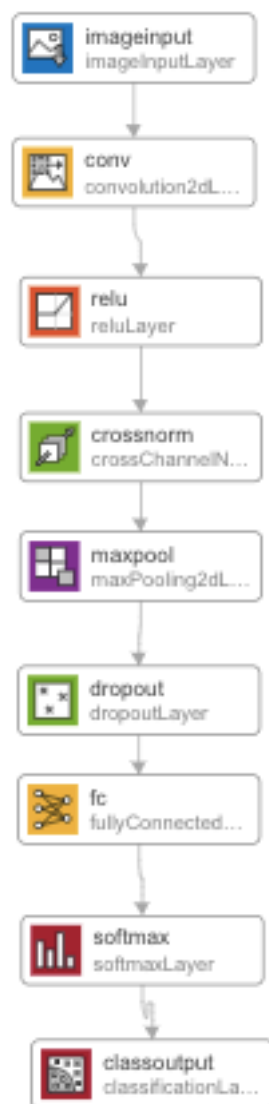
classificationLayer

Name: classoutput

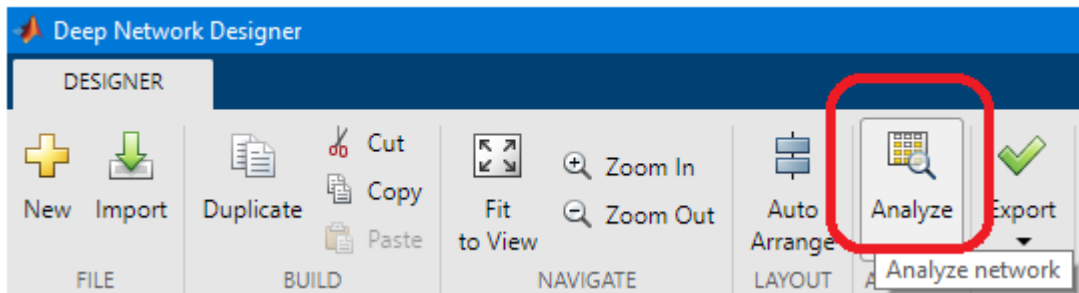
Classes: auto

OutputSize: auto

LossFunction: crossentropyex



Создав сеть и связав блоки стрелками, надо проверить согласованность соединений. Для этого выбираем пункт анализа сети.



Результат анализа

Deep Learning Network Analyzer

Network from Deep Network Designer

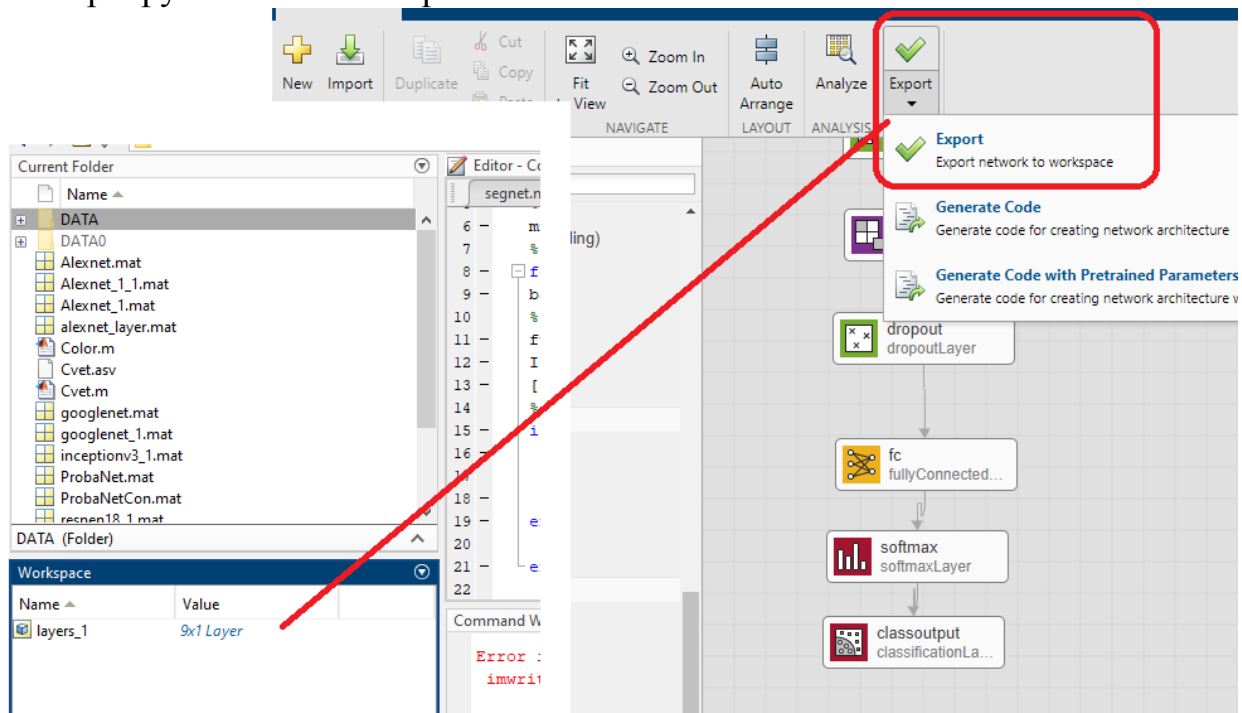
Analysis date: 25-Mar-2022 14:03:35

9 layers 0 warnings 0 errors

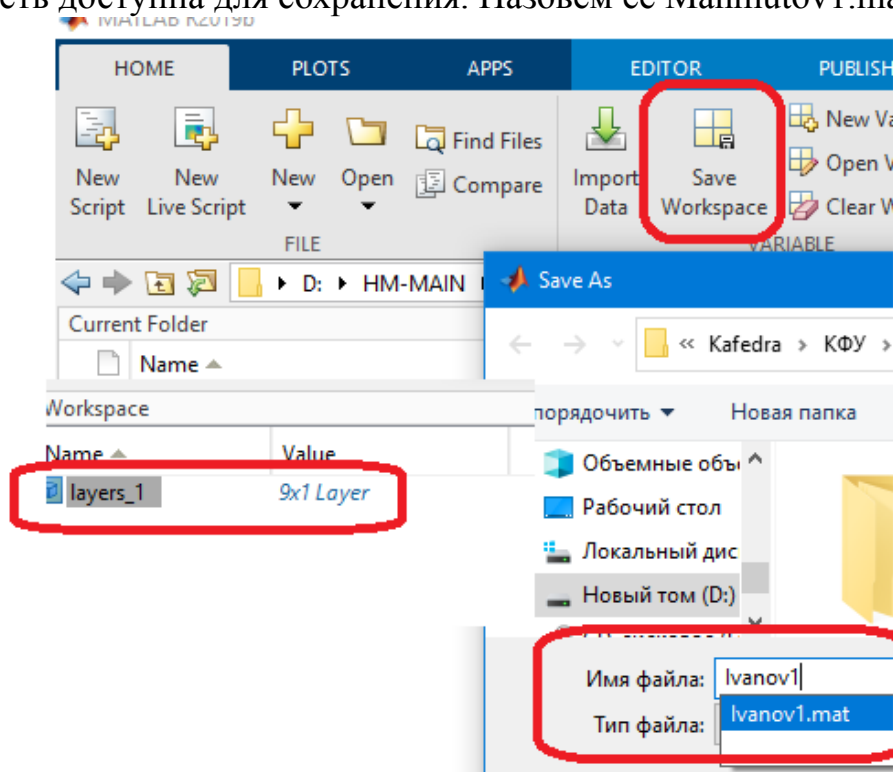
ANALYSIS RESULT

| | Name | Type | Activations | Learnables |
|---|--|-----------------------|-------------|---------------------------------|
| 1 | imageinput 227×227×3 images with 'z... | Image Input | 227×227×3 | – |
| 2 | conv 32 3×3 convolutions with st... | Convolution | 227×227×32 | Weights 3×3×3×32 Bias 1×1×32 |
| 3 | relu ReLU | ReLU | 227×227×32 | – |
| 4 | crossnorm cross channel normalizatio... | Cross Channel Nor... | 227×227×32 | – |
| 5 | maxpool 5×5 max pooling with strid... | Max Pooling | 227×227×32 | – |
| 6 | dropout 50% dropout | Dropout | 227×227×32 | – |
| 7 | fc 2 fully connected layer | Fully Connected | 1×1×2 | Weights 2×1648928 Bias 2×1 |
| 8 | softmax softmax | Softmax | 1×1×2 | – |
| 9 | classoutput crossentropyex | Classification Output | – | – |

Обратим внимание, что сеть сформирована **без ошибок**.
Экспортируем сеть в workspace.



Обратим внимание, что сеть экспортировалась с именем `layers_1` и под этим именем она будет «видна» в использующих ее программах. Из workspace созданная сеть доступна для сохранения. Назовем ее `Mahmutov1.mat`.



Обучим сформированную сеть.

Предварительно подготовим данные в требуемом формате. По умолчанию входные изображения у нас должны быть трехканальные. Преобразуем в директориях YES и NO одноканальные изображения в трехканальные с помощью программы Color.m , которая в заданной папке заменяет одноканальные изображения на трехканальные под тем же именем.

Текст программы:

```
%Преобразование одноканальных изображений в трехканальные в выбранном
%каталоге
% получаем директорию
myDir = uigetdir;
% только изображения bmp
myFiles = dir(fullfile(myDir, '*.bmp'));
% Пробегаемся по всем изображениям
for k = 1:length(myFiles)
baseFileName = myFiles(k).name; % имена файлов в каталоге myDir
% добавляем полный путь к имени файла
fullFileName = fullfile(myDir, baseFileName);
I = imread(fullFileName); % считываем файл изображения
[rows, columns, numberOfColorChannels] = size(I); % определяем размер
изображения изображения
% Применяем к изображению фильтр
if numberOfColorChannels == 1
    rgbimg=cat(3,I,I,I); % преобразование изображения из одноканального в
    трехканальное

imwrite(rgbimg,fullFileName); % сохранение изображения с тем же именем в том же
каталоге
end
end
```

Обучение сети произведем с помощью программы Lean_net.

Текст программы

```
clear all
load('Mahmutov1.mat'); %загружаем сеть
%загружаем данные
imds =
imageDatastore('DATA','IncludeSubfolders',true,'LabelSource','foldernames');
%делим данные на обучающую и проверочную выборки
[imdsTrain,imdsValidation] = splitEachLabel(imds,0.7,'randomized');
N = 227; % размер изображения для сети
%данные из обоих выборок приводим к одному формату
augimdsTrain = augmentedImageDatastore([N N],imdsTrain);
augimdsValidation = augmentedImageDatastore([N N],imdsValidation);

%задаем опции обучения и проверки
options = trainingOptions('adam', ... %стохастический градиентный спуск
    'MiniBatchSize',20, ... %кол-во изображений в каждой итерации
    'MaxEpochs',10, ... %кол-во эпох
    'InitialLearnRate',3e-5, ...
    'Shuffle','once', ... %перемешивание данных
    'ValidationData',augimdsValidation, ... %данные для проверки
    'ValidationFrequency',5, ... %частота проверки сети
    'Verbose',false, ... %вывод данных в ком. строку
    'ExecutionEnvironment','cpu', ... %аппаратные ресурсы
    'Plots','training-progress');

%обучение сети
netTransfer = trainNetwork(augimdsTrain, layers_1, options);
```

```

%проверка сети на точность
[YPred,probs] = classify(netTransfer,augimdsValidation);

A = [YPred { }];%метки на выходе сети
B = imdsValidation.Labels;%исходные метки
N = size(A,1);
for k = 1:N
if string([A(k,1)]) == "NO"
    C(k)= 0 ;% метке NO присваиваем значение 0
else
    C(k) = 1; % метке YES присваиваем значение 0
    % сохранение изображения с тем же именем в том же каталоге
end
if string([B(k,1)]) == "NO" % аналогично для исходных меток
else
    D(k) = 1;
end
end
%Определяем TP,TN,FP,FN

TP=0; TN=0; FP=0; FN=0;
for k = 1:N
    TP = TP+C(k)*D(k);
    TN = TN+(1-C(k))*(1-D(k));
    FP = FP+C(k)*(1-D(k));
    FN = FN+(1-C(k))*D(k);
end
%Определяем чувствительность, специфичность,точность
sensityve = TP/(TP+FN);
spesifice = TN/(TN+FP);
TP+FN
TN+FP
accuracy = mean(YPred == imdsValidation.Labels)
sensityve
spesifice

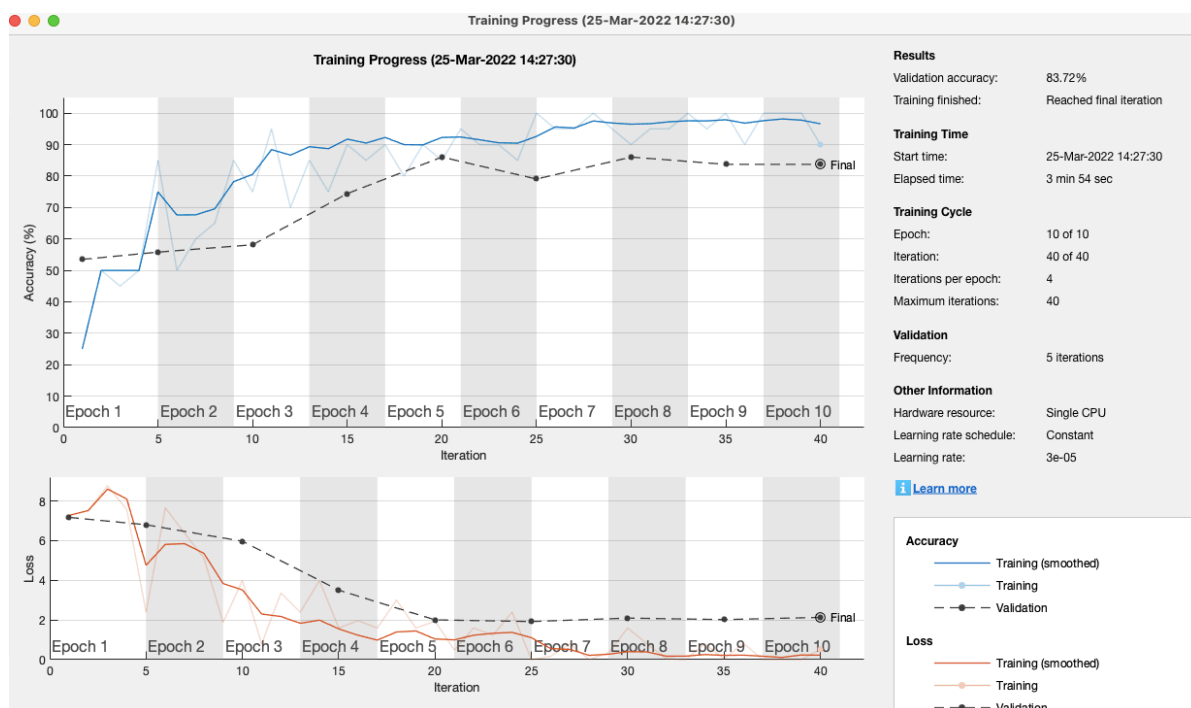
```

Результаты обучения.

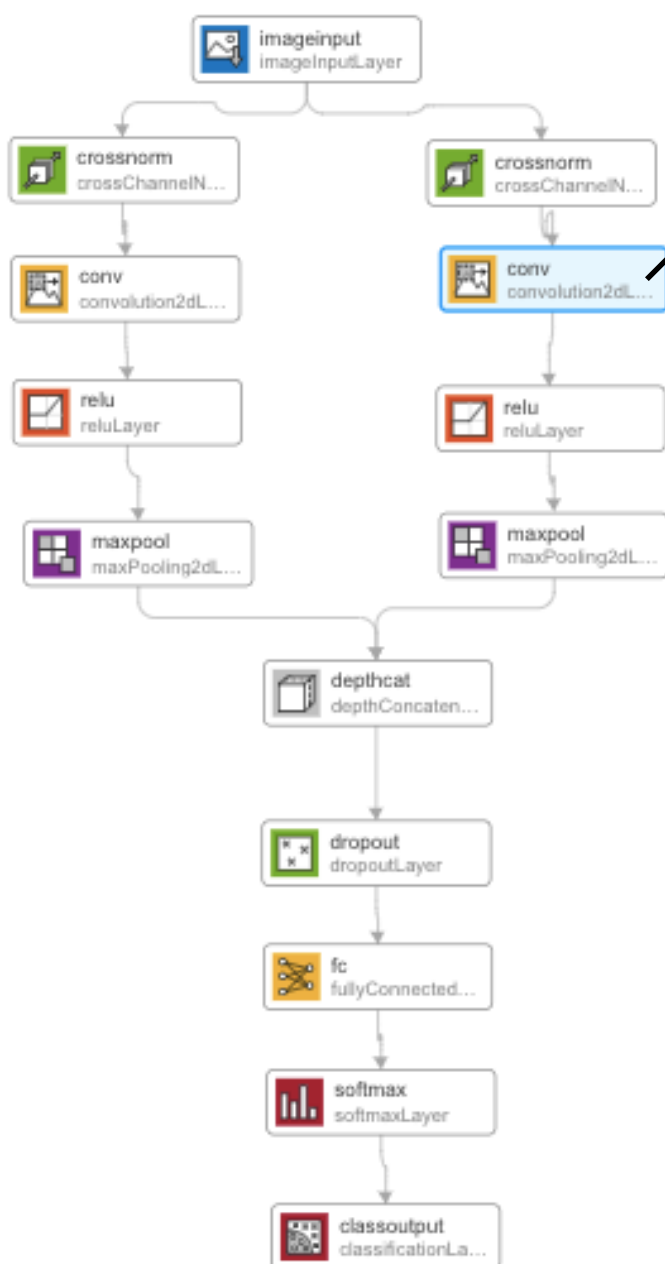
accuracy = 0.8372

sensityve = 0.8696

spesifice = 0.8000



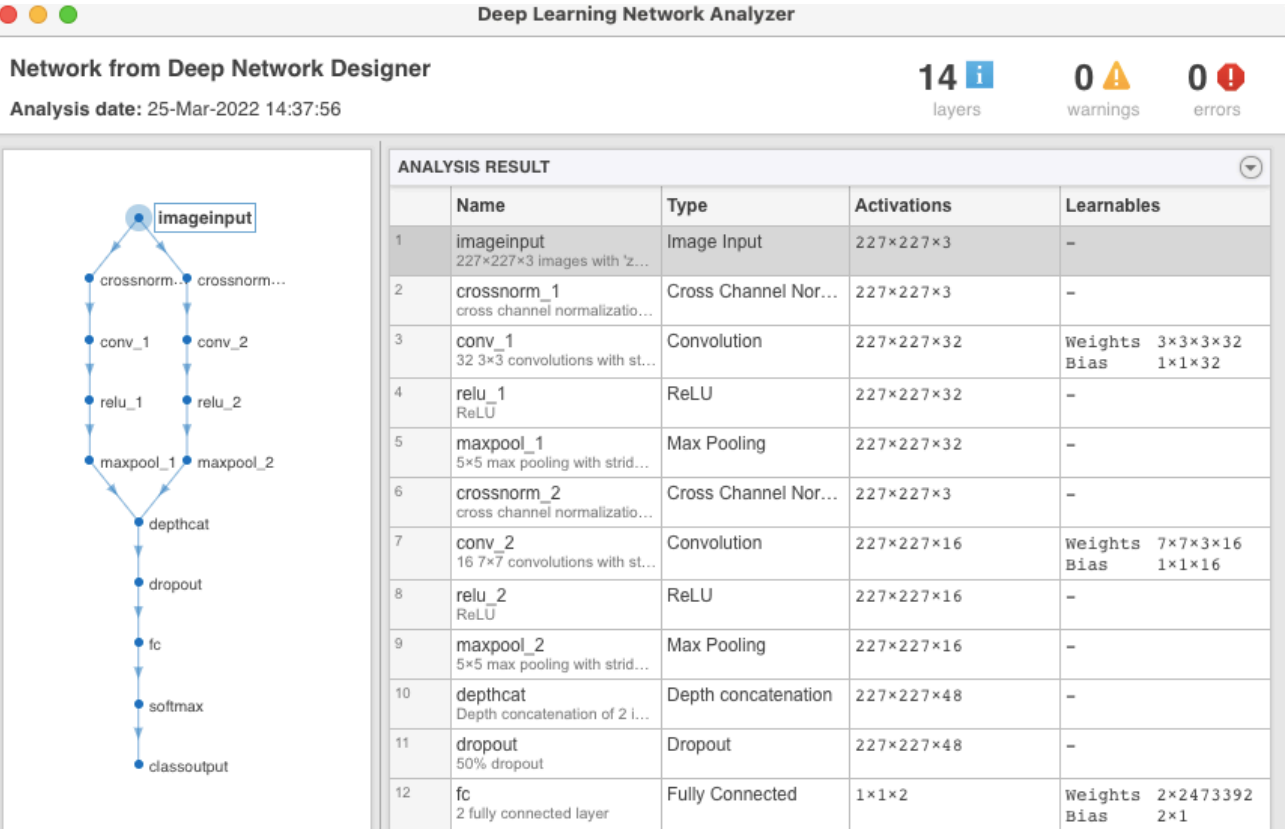
2) Создадим разветвленную сеть с двумя ветками свертки, как показано на рисунке ниже.



convolution2dLayer ?

| | |
|-----------------------|--------|
| Name | conv |
| FilterSize | 7,7 |
| NumFilters | 16 |
| Stride | 1,1 |
| DilationFactor | 1,1 |
| Padding | same |
| Weights | [] |
| Bias | [] |
| WeightLearnRateFactor | 1 |
| WeightL2Factor | 1 |
| BiasLearnRateFactor | 1 |
| BiasL2Factor | 0 |
| WeightsInitializer | glorot |
| BiasInitializer | zeros |

В левой ветке все параметры слоев взяты по умолчанию, а в правой ветке слой свертки изменен – используются 16 фильтров 7x7.
Результат анализа сети:



Сеть экспортировалась в workspace под именем lgraph_1. Сохраним ее в файле Mahmutov2.m .

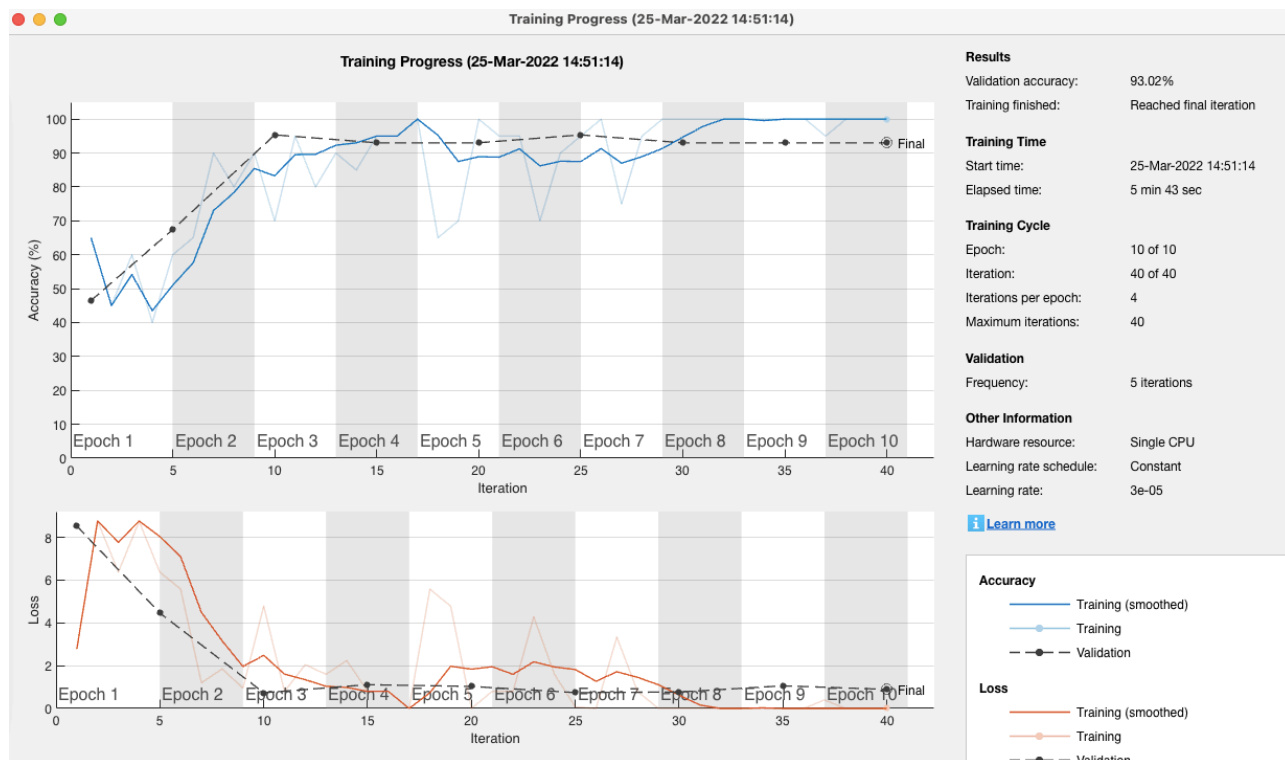
Обучение сети произведем с помощью программы Lean_net (в программе необходимо изменить имя сети), используя те же данные, что и в пункте 1).

Изменяется также имя сети:

%обучение сети

```
netTransfer = trainNetwork(augimdsTrain, lgraph_1, options);
```

Результаты обучения.



accuracy = 0.9302

Sensitivity, TPR = 0.9130

Specificity, TNR = 0.9500

Используя методы машинного обучения, чтобы добиться большого процента точности, надо было проделать манипуляции с данными(извлечение признаков Харалика, повышение контрастности, отбор признаков).

В глубоком обучении модель смогла сама извлечь значимые признаки. Последовательная сеть показала неплохие результаты. А разветвленная сеть с несколькими сверточными слоями с различными размерами фильтров дала более точные результаты. Также данная модель больше определяет положительных значений из общего числа истинно-положительных и больше определяет отрицательных значений из общего числа истинно-отрицательных.

| | accuracy | Sensitivity, TPR | Specificity, TNR |
|-------------------------------------|----------|------------------|------------------|
| I CNN | 0,8372 | 0,8696 | 0,800 |
| II CNN | 0,9302 | 0,9130 | 0,9500 |
| 1 задание(модель 11) | 0,91 | 0,9390 | 0,894 |
| 2 задание(отб признаков, модель 24) | 0,9226 | 0,9600 | 0,8900 |
| 3 задание (модель 11) | 0,9110 | 0,9390 | 0,8630 |