# Assignment-7

**Sai sahithi katuri**

**Links that I have gone through:**

<https://www.mathworks.com/help/vision/examples/object-detection-using-deep-learning.html>

<https://www.mathworks.com/help/vision/examples/semantic-segmentation-using-deep-learning.html>

<https://www.mathworks.com/help/vision/object-detection-using-deep-learning.html>

<https://www.mathworks.com/videos/deep-learning-in-11-lines-of-matlab-code-1481229977318.html>

**Site that I have downloaded code and data :**

<https://www.mathworks.com/help/vision/examples/semantic-segmentation-using-deep-learning.html>

**Extra things that I installed:**

I have installed ResNet-18 and it is an efficient network that is well suited for applications with limited processing resources.

And I have also used Deeplab v3+ which is one type of convolutional neural network (CNN) designed for semantic image segmentation

Used

**CODE:**

resnet18();

pretrainedURL = 'https://www.mathworks.com/supportfiles/vision/data/deeplabv3plusResnet18CamVid.mat';

pretrainedFolder = fullfile(tempdir,'pretrainedNetwork');

pretrainedNetwork = fullfile(pretrainedFolder,'deeplabv3plusResnet18CamVid.mat');

if ~exist(pretrainedNetwork,'file')

mkdir(pretrainedFolder);

disp('Downloading pretrained network (58 MB)...');

websave(pretrainedNetwork,pretrainedURL);

end

imageURL = 'http://web4.cs.ucl.ac.uk/staff/g.brostow/MotionSegRecData/files/701\_StillsRaw\_full.zip';

labelURL = 'http://web4.cs.ucl.ac.uk/staff/g.brostow/MotionSegRecData/data/LabeledApproved\_full.zip';

outputFolder = fullfile(tempdir,'CamVid');

labelsZip = fullfile(outputFolder,'labels.zip');

imagesZip = fullfile(outputFolder,'images.zip');

if ~exist(labelsZip, 'file') || ~exist(imagesZip,'file')

mkdir(outputFolder)

disp('Downloading 16 MB CamVid dataset labels...');

websave(labelsZip, labelURL);

unzip(labelsZip, fullfile(outputFolder,'labels'));

disp('Downloading 557 MB CamVid dataset images...');

websave(imagesZip, imageURL);

unzip(imagesZip, fullfile(outputFolder,'images'));

end

imgDir = fullfile(outputFolder,'images','701\_StillsRaw\_full');

imds = imageDatastore(imgDir);

I = readimage(imds,1);

I = histeq(I);

imshow(I)

classes = [

"Sky"

"Building"

"Pole"

"Road"

"Pavement"

"Tree"

"SignSymbol"

"Fence"

"Car"

"Pedestrian"

"Bicyclist"

];

labelIDs = camvidPixelLabelIDs();

labelDir = fullfile(outputFolder,'labels');

pxds = pixelLabelDatastore(labelDir,classes,labelIDs);

C = readimage(pxds,1);

cmap = camvidColorMap;

B = labeloverlay(I,C,'ColorMap',cmap);

imshow(B)

pixelLabelColorbar(cmap,classes);

tbl = countEachLabel(pxds)

frequency = tbl.PixelCount/sum(tbl.PixelCount);

bar(1:numel(classes),frequency)

xticks(1:numel(classes))

xticklabels(tbl.Name)

xtickangle(45)

ylabel('Frequency')

[imdsTrain, imdsVal, imdsTest, pxdsTrain, pxdsVal, pxdsTest] = partitionCamVidData(imds,pxds);

numTrainingImages = numel(imdsTrain.Files)

numValImages = numel(imdsVal.Files)

numTestingImages = numel(imdsTest.Files)

% Specify the network image size. This is typically the same as the traing image sizes.

imageSize = [720 960 3];

% Specify the number of classes.

numClasses = numel(classes);

% Create DeepLab v3+.

lgraph = deeplabv3plusLayers(imageSize, numClasses, "resnet18");

imageFreq = tbl.PixelCount ./ tbl.ImagePixelCount;

classWeights = median(imageFreq) ./ imageFreq

pxLayer = pixelClassificationLayer('Name','labels','Classes',tbl.Name,'ClassWeights',classWeights);

lgraph = replaceLayer(lgraph,"classification",pxLayer);

% Define validation data.

pximdsVal = pixelLabelImageDatastore(imdsVal,pxdsVal);

% Define training options.

options = trainingOptions('sgdm', ...

'LearnRateSchedule','piecewise',...

'LearnRateDropPeriod',10,...

'LearnRateDropFactor',0.3,...

'Momentum',0.9, ...

'InitialLearnRate',1e-3, ...

'L2Regularization',0.005, ...

'ValidationData',pximdsVal,...

'MaxEpochs',30, ...

'MiniBatchSize',8, ...

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

'CheckpointPath', tempdir, ...

'VerboseFrequency',2,...

'Plots','training-progress',...

'ValidationPatience', 4);

augmenter = imageDataAugmenter('RandXReflection',true,...

'RandXTranslation',[-10 10],'RandYTranslation',[-10 10]);

pximds = pixelLabelImageDatastore(imdsTrain,pxdsTrain, ...

'DataAugmentation',augmenter);

doTraining = false;

if doTraining

[net, info] = trainNetwork(pximds,lgraph,options);

else

data = load(pretrainedNetwork);

net = data.net;

end

I = readimage(imdsTest,35);

C = semanticseg(I, net);

B = labeloverlay(I,C,'Colormap',cmap,'Transparency',0.4);

imshow(B)

pixelLabelColorbar(cmap, classes);

expectedResult = readimage(pxdsTest,35);

actual = uint8(C);

expected = uint8(expectedResult);

imshowpair(actual, expected)

iou = jaccard(C,expectedResult);

table(classes,iou)

pxdsResults = semanticseg(imdsTest,net, ...

'MiniBatchSize',4, ...

'WriteLocation',tempdir, ...

'Verbose',false);

metrics = evaluateSemanticSegmentation(pxdsResults,pxdsTest,'Verbose',false);

metrics.DataSetMetrics

metrics.ClassMetrics

function labelIDs = camvidPixelLabelIDs()

% Return the label IDs corresponding to each class.

%

% The CamVid dataset has 32 classes. Group them into 11 classes following

% the original SegNet training methodology [1].

%

% The 11 classes are:

% "Sky" "Building", "Pole", "Road", "Pavement", "Tree", "SignSymbol",

% "Fence", "Car", "Pedestrian", and "Bicyclist".

%

% CamVid pixel label IDs are provided as RGB color values. Group them into

% 11 classes and return them as a cell array of M-by-3 matrices. The

% original CamVid class names are listed alongside each RGB value. Note

% that the Other/Void class are excluded below.

labelIDs = { ...

% "Sky"

[

128 128 128; ... % "Sky"

]

% "Building"

[

000 128 064; ... % "Bridge"

128 000 000; ... % "Building"

064 192 000; ... % "Wall"

064 000 064; ... % "Tunnel"

192 000 128; ... % "Archway"

]

% "Pole"

[

192 192 128; ... % "Column\_Pole"

000 000 064; ... % "TrafficCone"

]

% Road

[

128 064 128; ... % "Road"

128 000 192; ... % "LaneMkgsDriv"

192 000 064; ... % "LaneMkgsNonDriv"

]

% "Pavement"

[

000 000 192; ... % "Sidewalk"

064 192 128; ... % "ParkingBlock"

128 128 192; ... % "RoadShoulder"

]

% "Tree"

[

128 128 000; ... % "Tree"

192 192 000; ... % "VegetationMisc"

]

% "SignSymbol"

[

192 128 128; ... % "SignSymbol"

128 128 064; ... % "Misc\_Text"

000 064 064; ... % "TrafficLight"

]

% "Fence"

[

064 064 128; ... % "Fence"

]

% "Car"

[

064 000 128; ... % "Car"

064 128 192; ... % "SUVPickupTruck"

192 128 192; ... % "Truck\_Bus"

192 064 128; ... % "Train"

128 064 064; ... % "OtherMoving"

]

% "Pedestrian"

[

064 064 000; ... % "Pedestrian"

192 128 064; ... % "Child"

064 000 192; ... % "CartLuggagePram"

064 128 064; ... % "Animal"

]

% "Bicyclist"

[

000 128 192; ... % "Bicyclist"

192 000 192; ... % "MotorcycleScooter"

]

};

end

function pixelLabelColorbar(cmap, classNames)

% Add a colorbar to the current axis. The colorbar is formatted

% to display the class names with the color.

colormap(gca,cmap)

% Add colorbar to current figure.

c = colorbar('peer', gca);

% Use class names for tick marks.

c.TickLabels = classNames;

numClasses = size(cmap,1);

% Center tick labels.

c.Ticks = 1/(numClasses\*2):1/numClasses:1;

% Remove tick mark.

c.TickLength = 0;

end

function cmap = camvidColorMap()

% Define the colormap used by CamVid dataset.

cmap = [

128 128 128 % Sky

128 0 0 % Building

192 192 192 % Pole

128 64 128 % Road

60 40 222 % Pavement

128 128 0 % Tree

192 128 128 % SignSymbol

64 64 128 % Fence

64 0 128 % Car

64 64 0 % Pedestrian

0 128 192 % Bicyclist

];

% Normalize between [0 1].

cmap = cmap ./ 255;

end

function [imdsTrain, imdsVal, imdsTest, pxdsTrain, pxdsVal, pxdsTest] = partitionCamVidData(imds,pxds)

% Partition CamVid data by randomly selecting 60% of the data for training. The

% rest is used for testing.

% Set initial random state for example reproducibility.

rng(0);

numFiles = numel(imds.Files);

shuffledIndices = randperm(numFiles);

% Use 60% of the images for training.

numTrain = round(0.60 \* numFiles);

trainingIdx = shuffledIndices(1:numTrain);

% Use 20% of the images for validation

numVal = round(0.20 \* numFiles);

valIdx = shuffledIndices(numTrain+1:numTrain+numVal);

% Use the rest for testing.

testIdx = shuffledIndices(numTrain+numVal+1:end);

% Create image datastores for training and test.

trainingImages = imds.Files(trainingIdx);

valImages = imds.Files(valIdx);

testImages = imds.Files(testIdx);

imdsTrain = imageDatastore(trainingImages);

imdsVal = imageDatastore(valImages);

imdsTest = imageDatastore(testImages);

% Extract class and label IDs info.

classes = pxds.ClassNames;

labelIDs = camvidPixelLabelIDs();

% Create pixel label datastores for training and test.

trainingLabels = pxds.Files(trainingIdx);

valLabels = pxds.Files(valIdx);

testLabels = pxds.Files(testIdx);

pxdsTrain = pixelLabelDatastore(trainingLabels, classes, labelIDs);

pxdsVal = pixelLabelDatastore(valLabels, classes, labelIDs);

pxdsTest = pixelLabelDatastore(testLabels, classes, labelIDs);

end

**Description of code:**

Downloading the CamVid dataset from the given URLs.

We are Using  imageDatastore to load CamVid images.

Displaying one of the images

Using  pixelLabelDatastore to load CamVid pixel label image data.

The label statistics is verified

Created class weights and pixel weights

Deeplab v3+ is trained using 60% of the images from the dataset. The rest of the images are split evenly in 20% and 20% for validation and testing respectively.

Then created a network using lgraph

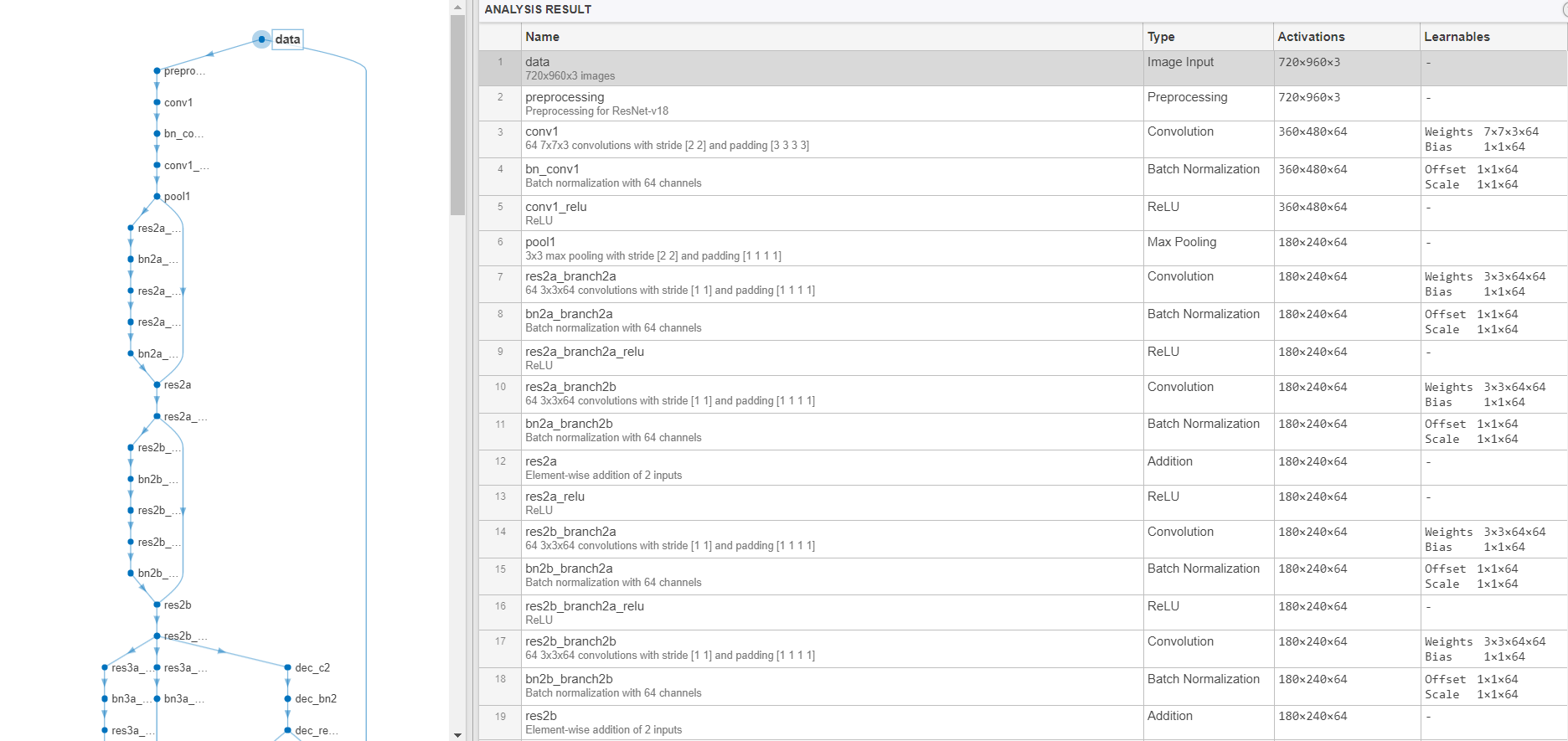
Seleted the training options

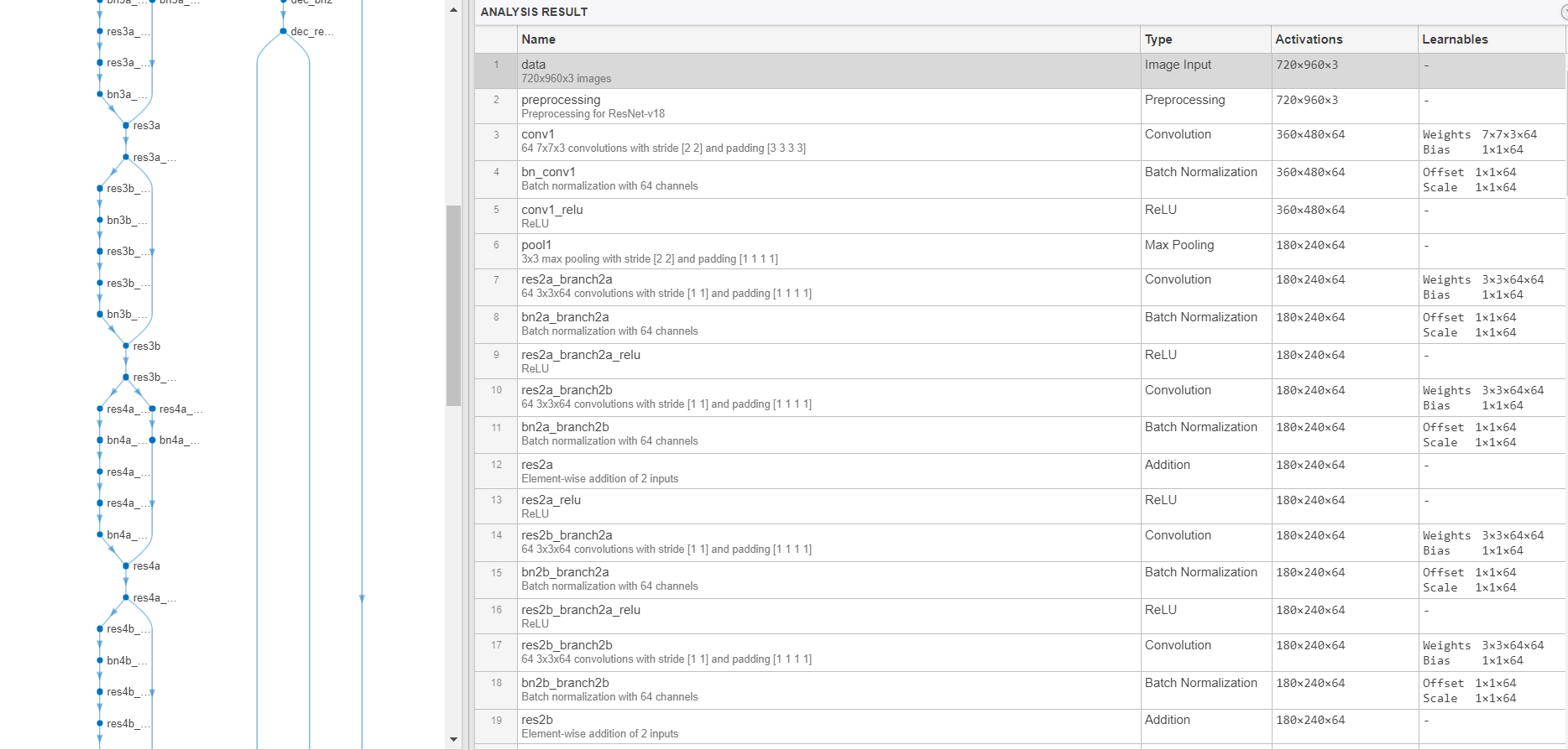
Replaced the layer with the pixel weights according to the dataset

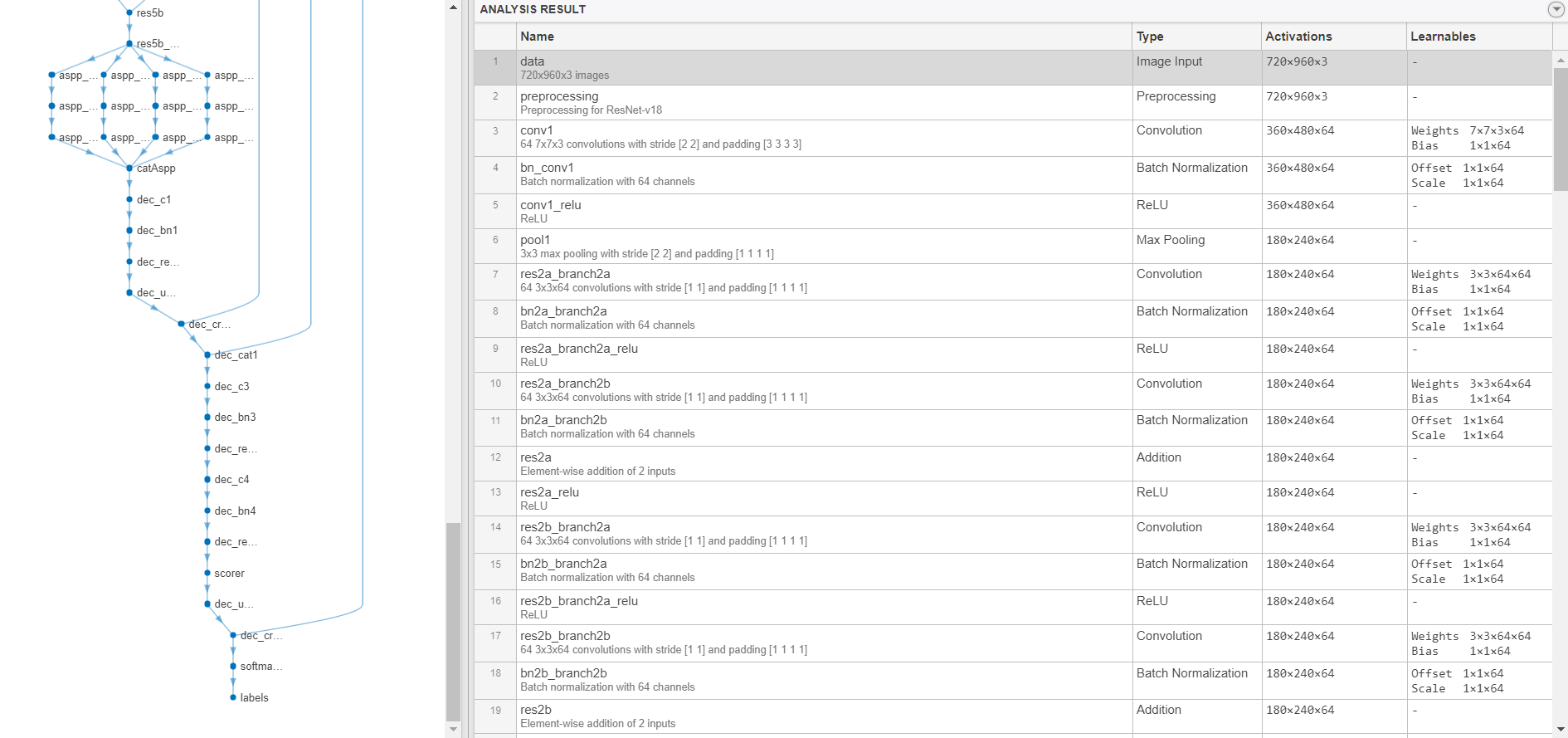
And test network on one image

To measure accuracy for multiple test images, runsemanticseg on the entire test set. A mini-batch size of 4 is used to reduce memory usage while segmenting images

**Architecture:**







Name PixelCount ImagePixelCount

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

{'Sky' } 7.6801e+07 4.8315e+08

{'Building' } 1.1737e+08 4.8315e+08

{'Pole' } 4.7987e+06 4.8315e+08

{'Road' } 1.4054e+08 4.8453e+08

{'Pavement' } 3.3614e+07 4.7209e+08

{'Tree' } 5.4259e+07 4.479e+08

{'SignSymbol'} 5.2242e+06 4.6863e+08

{'Fence' } 6.9211e+06 2.516e+08

{'Car' } 2.4437e+07 4.8315e+08

{'Pedestrian'} 3.4029e+06 4.4444e+08

{'Bicyclist' } 2.5912e+06 2.6196e+08

numTrainingImages =

421

numValImages =

140

numTestingImages =

140

classWeights =

0.3182

0.2082

5.0924

0.1744

0.7103

0.4175

4.5371

1.8386

1.0000

6.6059

5.1133

ans =

11×2 table

classes iou

\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_

"Sky" 0.91837

"Building" 0.84479

"Pole" 0.31203

"Road" 0.93698

"Pavement" 0.82838

"Tree" 0.89636

"SignSymbol" 0.57644

"Fence" 0.71046

"Car" 0.66688

"Pedestrian" 0.48417

"Bicyclist" 0.68431

