Pascal Triangle - Convolutions

This time let's look at what convolutional layers could achieve the same outcome as Pascal Triangle.

numLayers = 10;

for i = 1:numLayers

cl(i) = convolution2dLayer([2,2],1,"Bias",zeros(1,1,1),"Weights",[ones(1,2,1,1);zeros(1,2,1,1)],...

"Padding",[1 0 1 0],"Name","pasc"+int2str(i));

end

layers = [imageInputLayer([10 10 1],'Normalization',"none","Name","unit\_input")

cl.'

regressionLayer("Name","output")];

convPasc = assembleNetwork(layers);

inputs = zeros(10,10);

inputs(1,1)=1;

p = zeros(10,10,10);

for i = 1:10

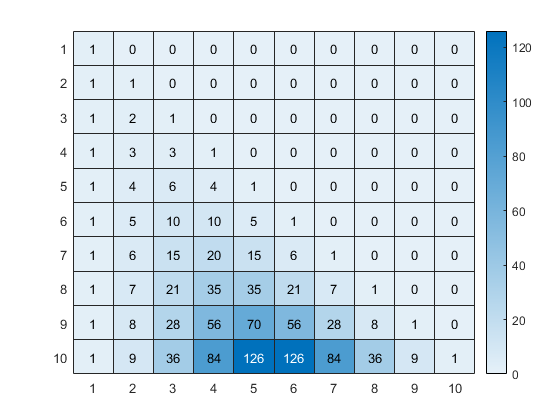
a = activations(convPasc,inputs,i);

p(:,:,i) = a;

end

out = sum(p,3);

heatmap(out)



bernoulilayers = [imageInputLayer([10 10 1],'Normalization',"none")

convolution2dLayer([1,10],10,"Bias",zeros(1,1,10),"Weights",reshape(eye(10),1,10,1,10),...

"Padding",[0 0 9 0],"Name","Bernoulli")

regressionLayer()];

convBern = assembleNetwork(bernoulilayers);

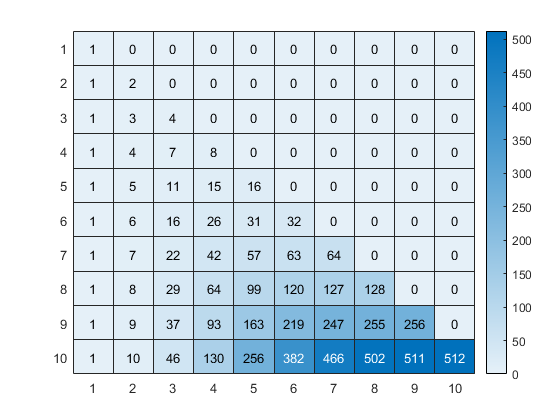
a = activations(convBern,out,2);

% a = activations(convBern,ones(10),2);

out2 = tril(sum(a,3));

figure

heatmap(out2)



lgpasc = layerGraph(layers(1:end-1));

connector = [additionLayer(10,"Name","Mux\_in")];%convolution2dLayer(1,1,"Bias",zeros(1,1,1,1),"Weights",ones(1,1,10,1),"Name","sumPascal")];

lgpasc = addLayers(lgpasc,connector);

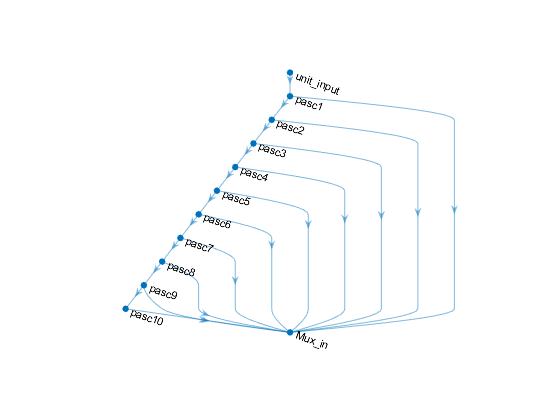
for i = 1:numel(cl)

lgpasc = connectLayers(lgpasc,"pasc"+int2str(i),"Mux\_in/in"+int2str(i));

end

plot(lgpasc);

axis off



Add Bernoulli Layers.

lgpasc = addLayers(lgpasc,bernoulilayers(2));

lgpasc = connectLayers(lgpasc,"Mux\_in","Bernoulli");

lgpasc = addLayers(lgpasc,convolution2dLayer(1,1,"Bias",zeros(1,1,1,1),"Weights",ones(1,1,10,1),"Name","sumBernoulli"));

lgpasc = connectLayers(lgpasc,"Bernoulli","sumBernoulli");

Now add back an output layer.

lgpasc = addLayers(lgpasc,depthConcatenationLayer(2,"Name","stack\_triangle"));

lgpasc = addLayers(lgpasc,layers(end));

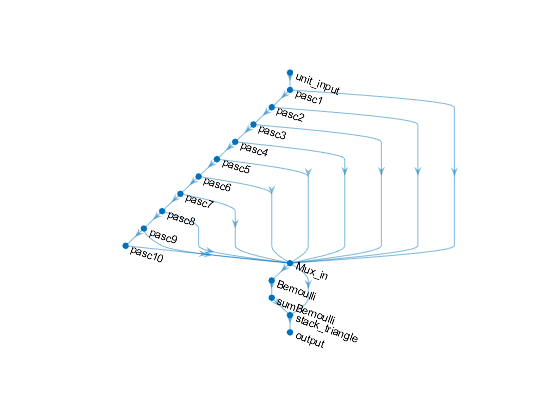
lgpasc = connectLayers(lgpasc,"Mux\_in","stack\_triangle/in1");

lgpasc = connectLayers(lgpasc,"sumBernoulli","stack\_triangle/in2");

lgpasc = connectLayers(lgpasc,"stack\_triangle","output");

plot(lgpasc);

axis off



convBern = assembleNetwork(lgpasc);

p = predict(convBern,inputs);

a = activations(convBern,inputs,'Bernoulli');