MCDB 4110/6440 – Quantitative Microscopy Lab

Lab 4 – 2:

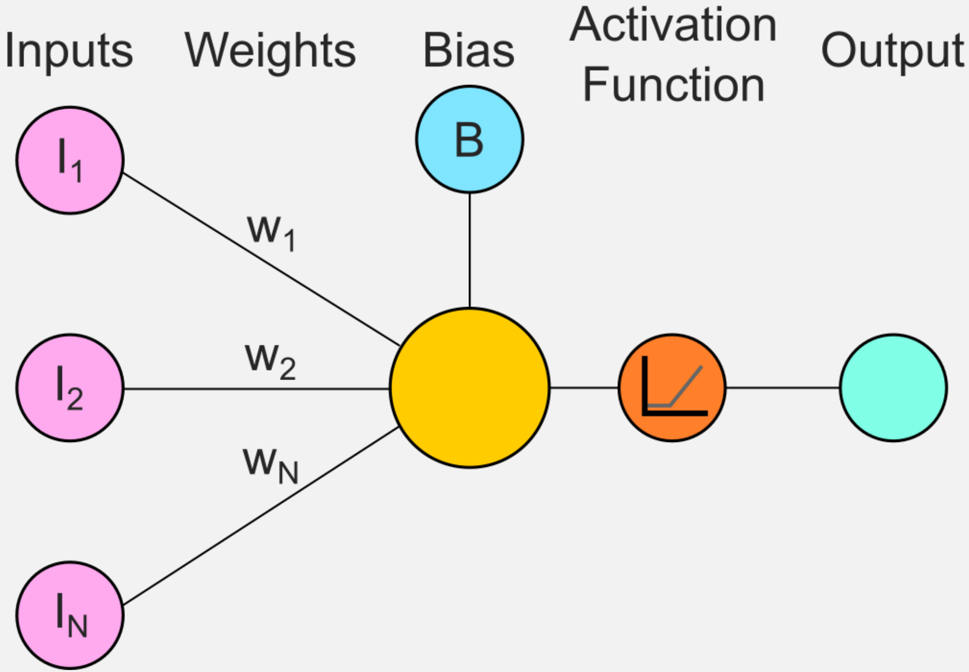
Evaluating a network and semantic segmentation

Lecturer: Jian Wei Tay

Date: 18 November 2021

Recap of last week

Activation function



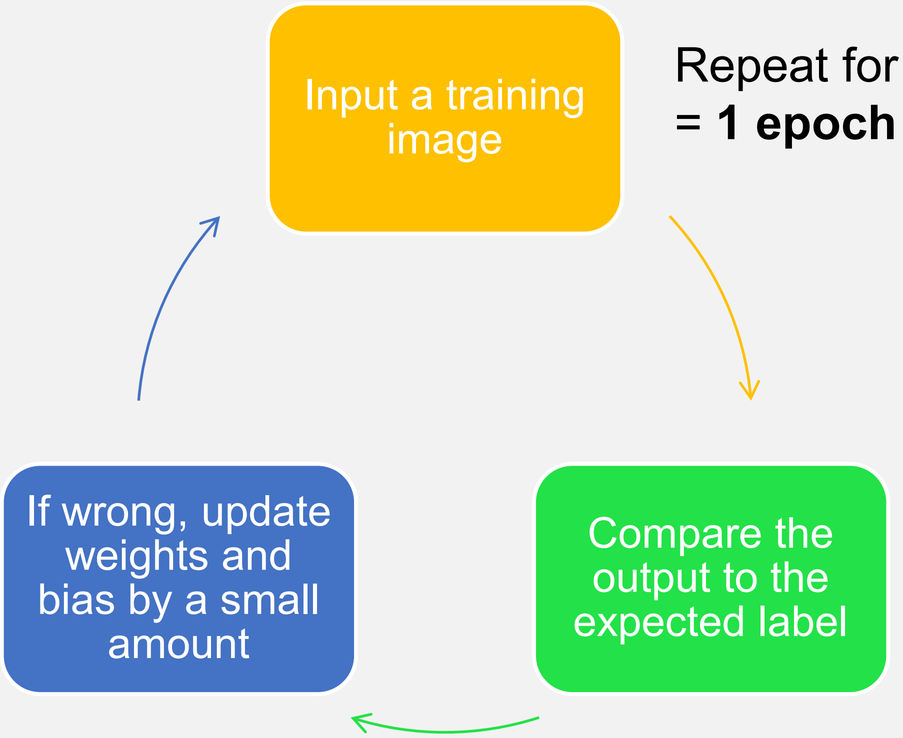
Step activation function

Output:

$= 1 \text{ if } \sum_{N}^{i=1} w_i I_i - B > 0$

$= 0$ otherwise

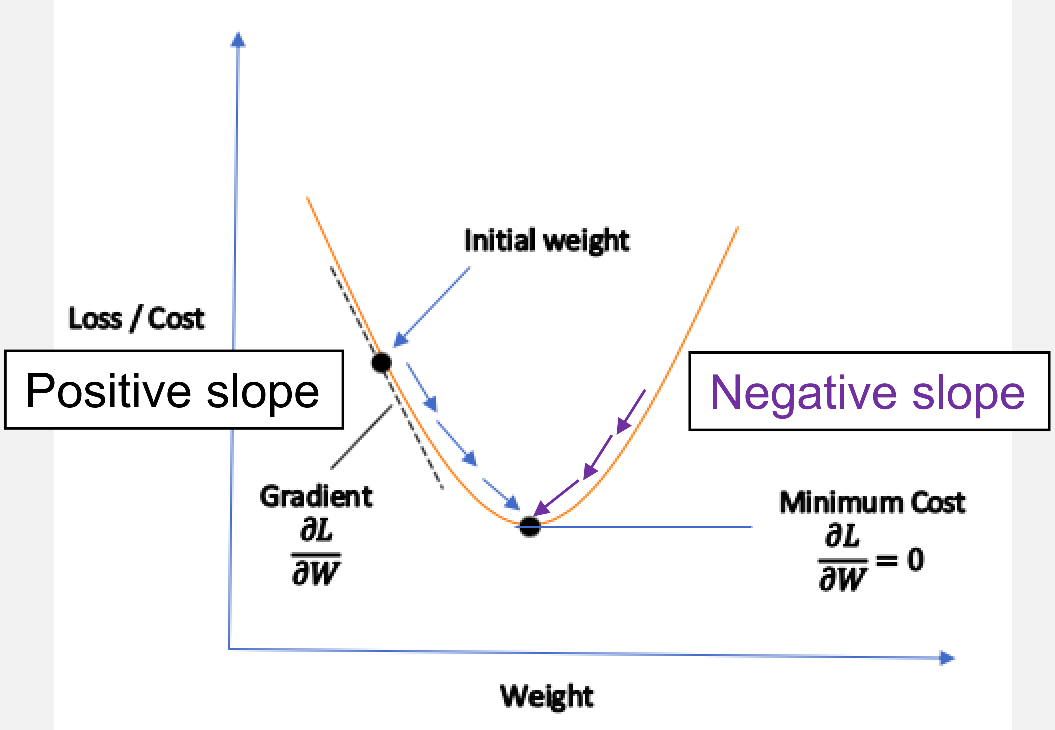
A typical training cycle



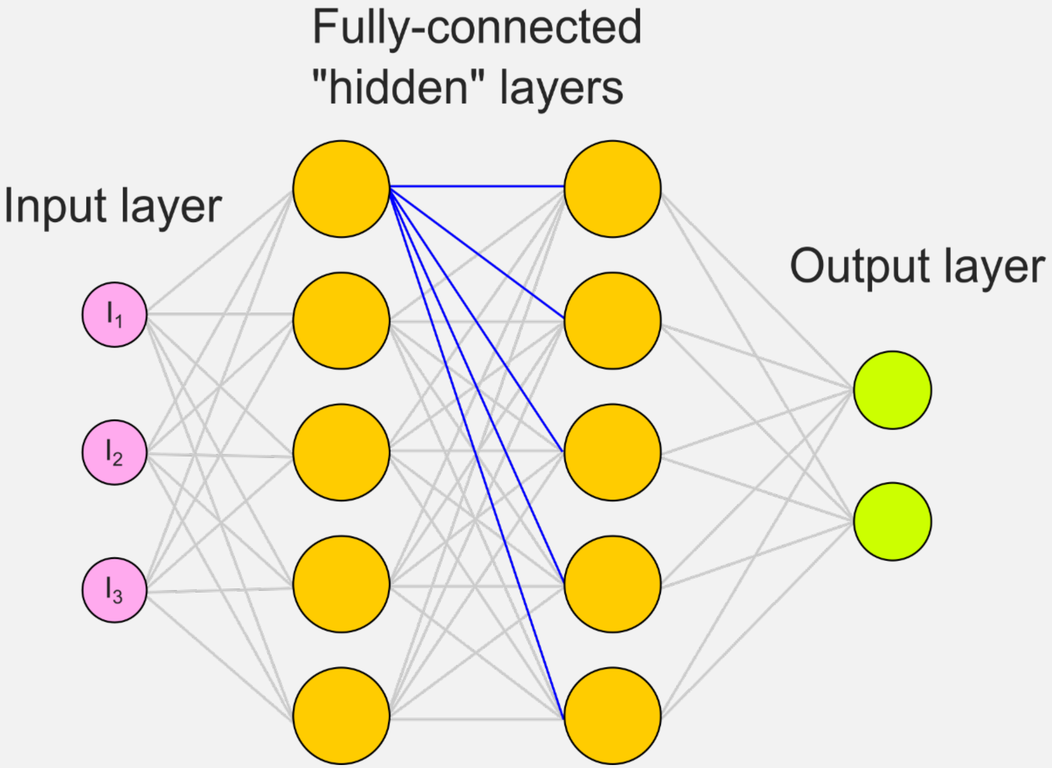
Repeat for entire set of training images
= **1 epoch**

Training runs for a set number of epochs decided by the user

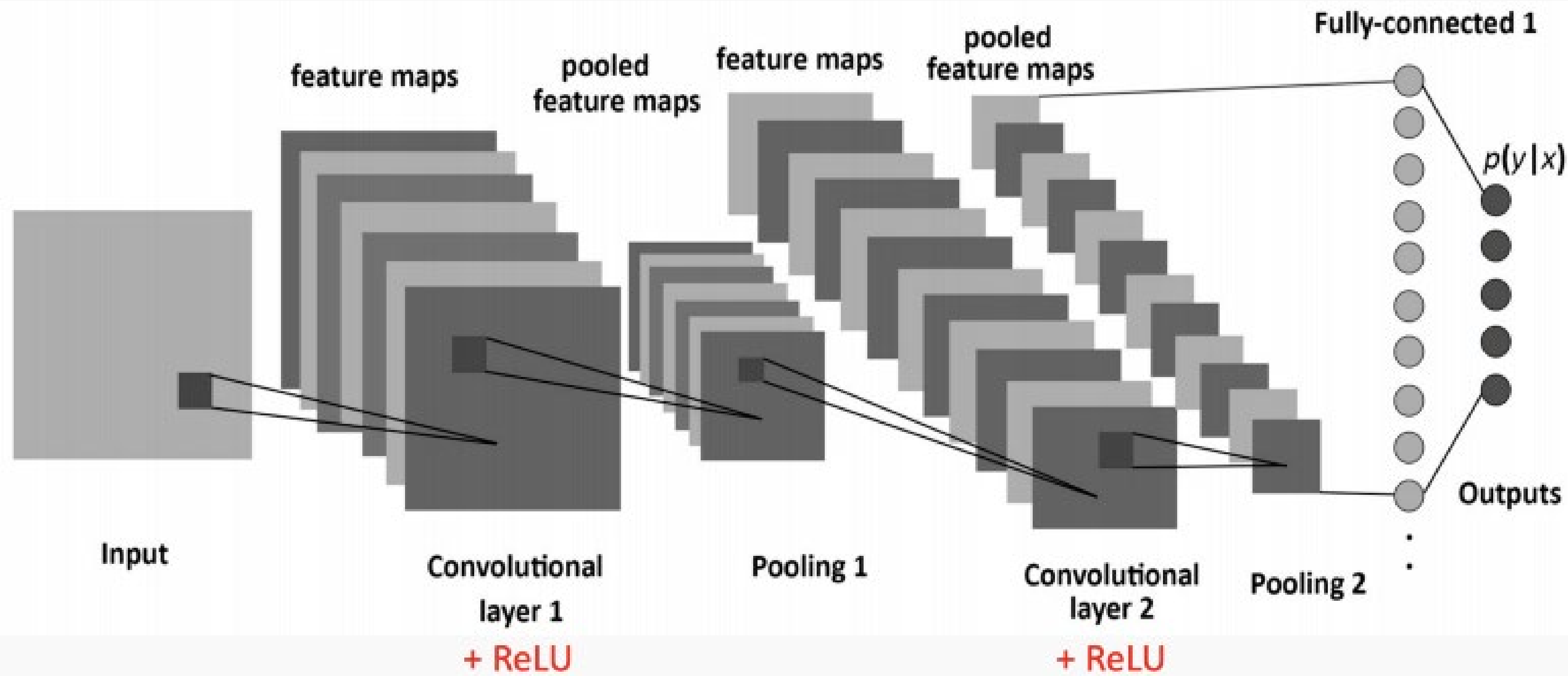
SGDM computes the derivative of the error, then changes the weights based on this slope



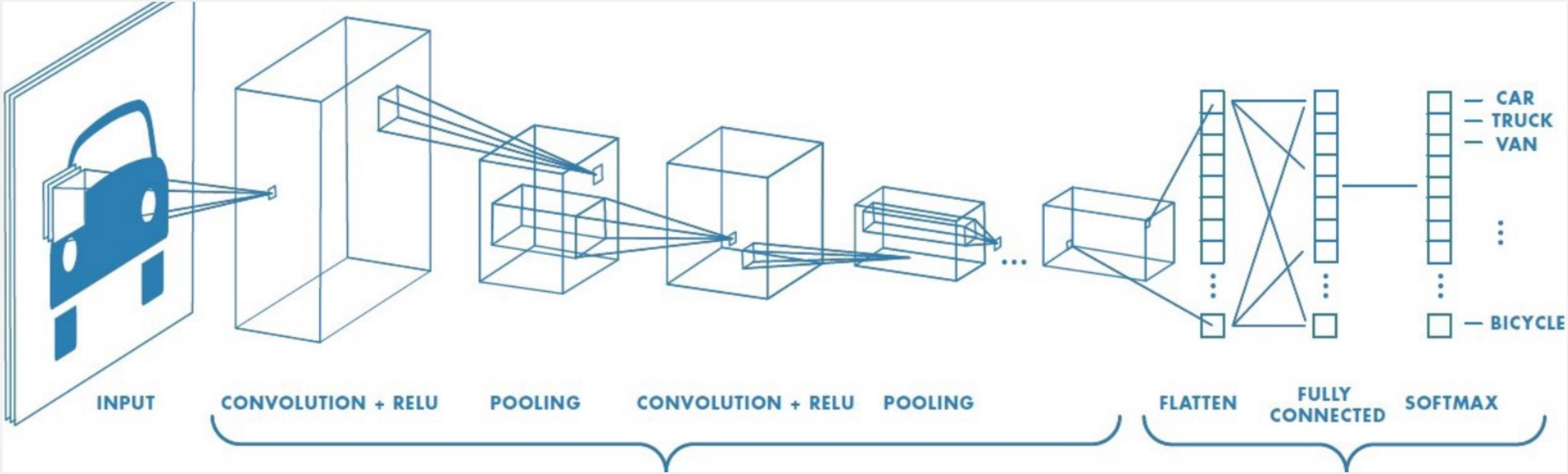
Neural networks have connected layers of perceptrons



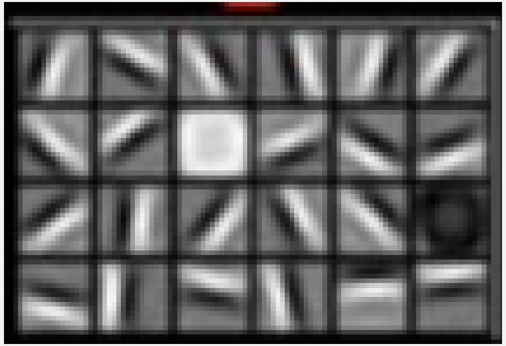
Pooling is used to combine pixels in the feature maps (e.g., combine 2x2 pixels into 1)



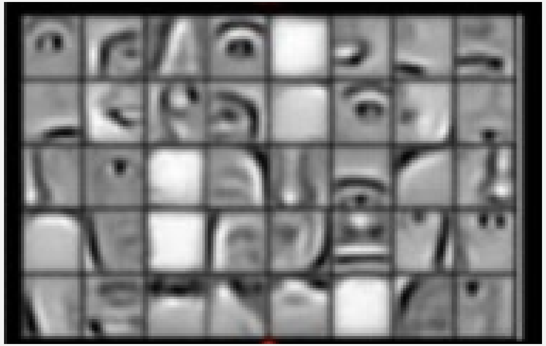
Convolutional neural networks have several layers of convolutional layers



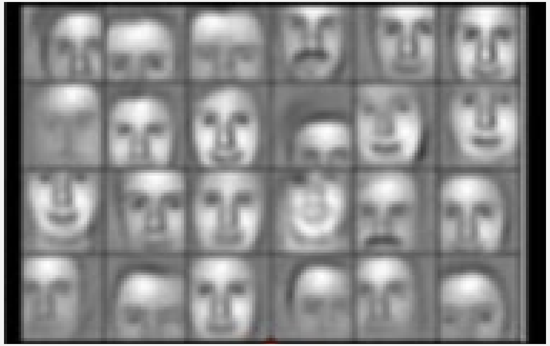
Each convolutional layer combines features from the previous



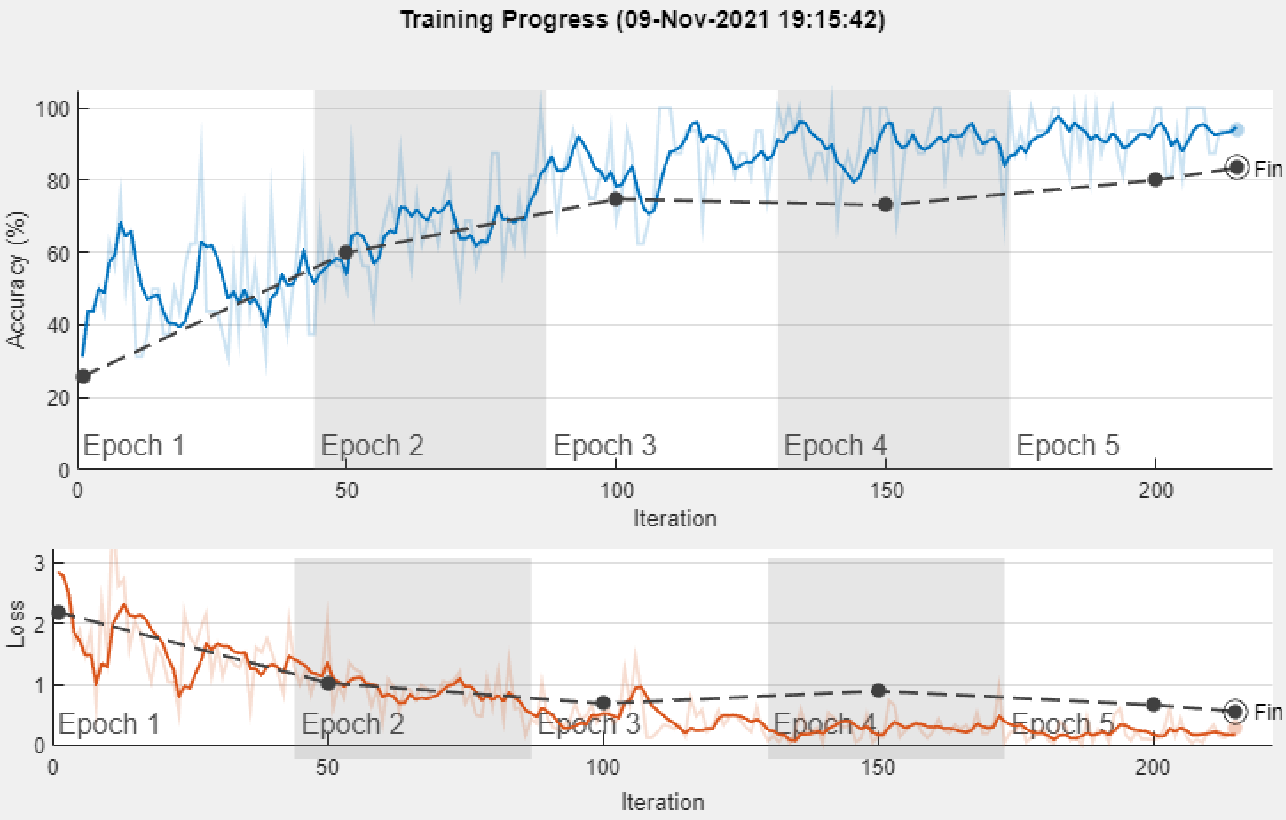
Layer 1
Detect lines and edges



Layer 2
Combine lines and edges to detect eyes, ears, noses



Layer 3
Combine eyes, ears, noses to detect faces



Results

Validation accuracy: 83.43%

Training finished: Reached final iteration

Training Time

Start time: 09-Nov-2021 19:15:42

Elapsed time: 55 sec

Training Cycle

Epoch: 5 of 5

Iteration: 215 of 215

Iterations per epoch: 43

Maximum iterations: 215

Validation

Frequency: 50 iterations

Other Information

Hardware resource: Single GPU

Learning rate schedule: Constant

Learning rate: 0.001

Learn more

pizza

hamburger

pizza

hamburger

pizza

pizza

hamburger

pizza

pizza

french_fries

pizza

pizza

hamburger

french_fries

hamburger

pizza

Deep Learning Visualization

Activations

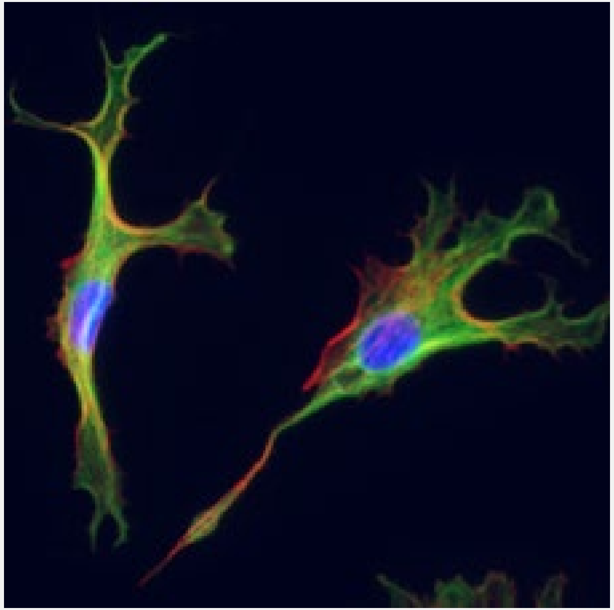
- To understand what filters have been trained, you can use the function `activations` to extract learned image features from a trained convolutional network

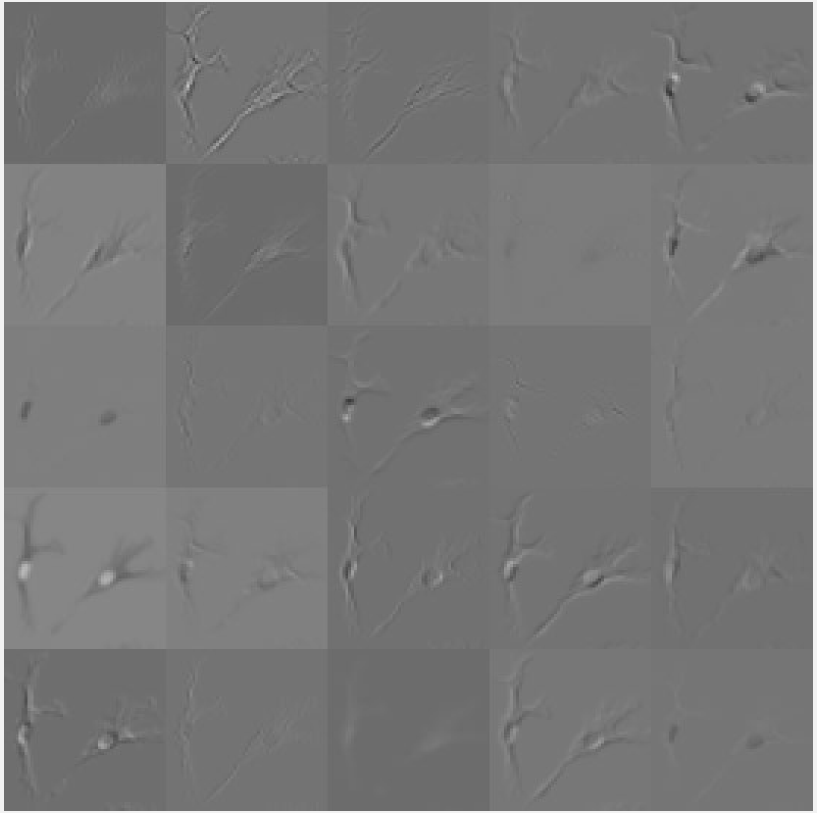
```
act = activations(network, image, layer)
```

Code

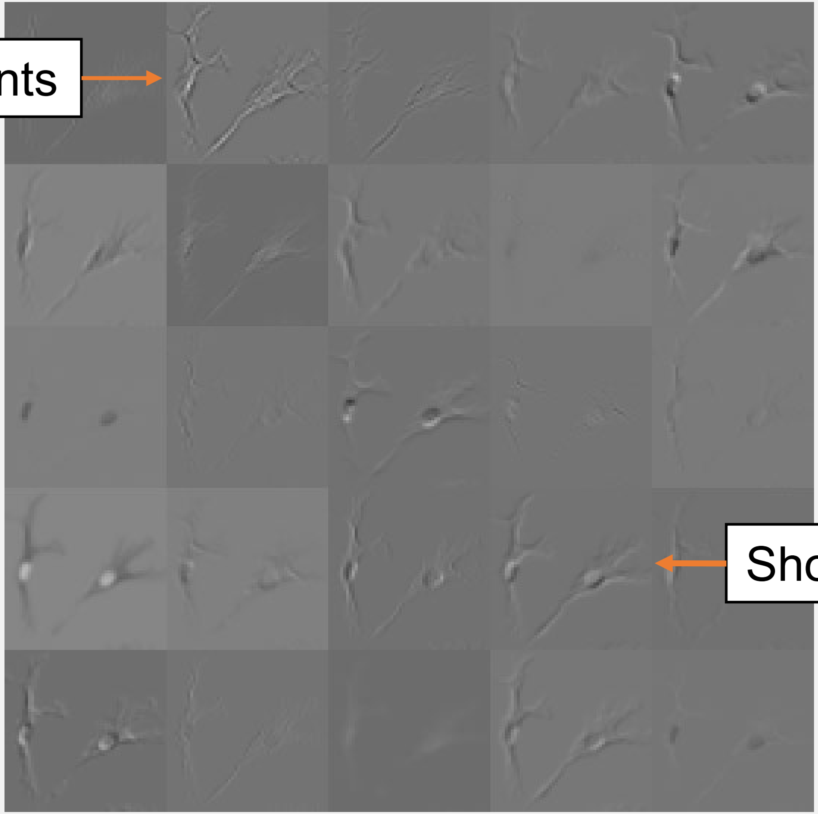
```
Irgb = imread('D:\Teaching\IQBioLabs_2021\Week 1\bpae_fullcell.tif');

act1 = activations(alexnet_lab4, Irgb, 'conv1');
sz = size(act1);
act1 = reshape(act1,[sz(1) sz(2) 1 sz(3)]);
I = imtile(mat2gray(act1),'GridSize',[5 5]);

imshow(I)
```

Example: First 25 activations from Layer 1





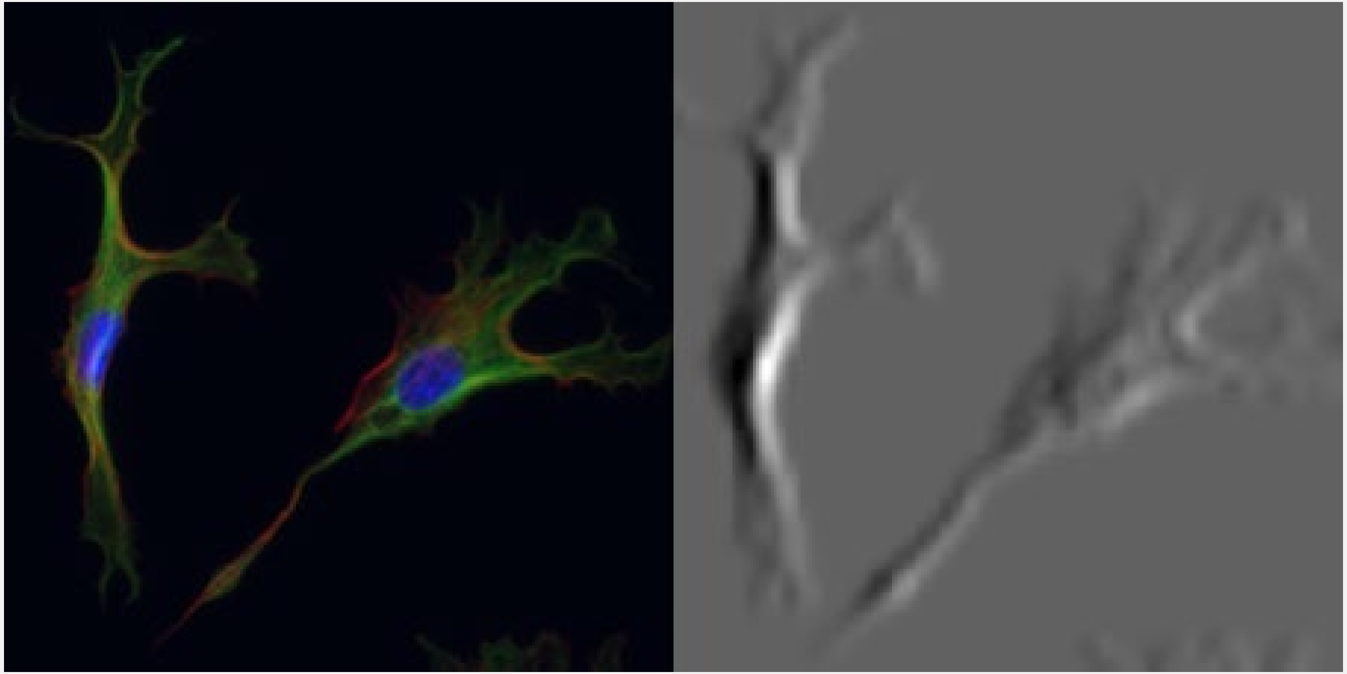
The activations show features that were detected



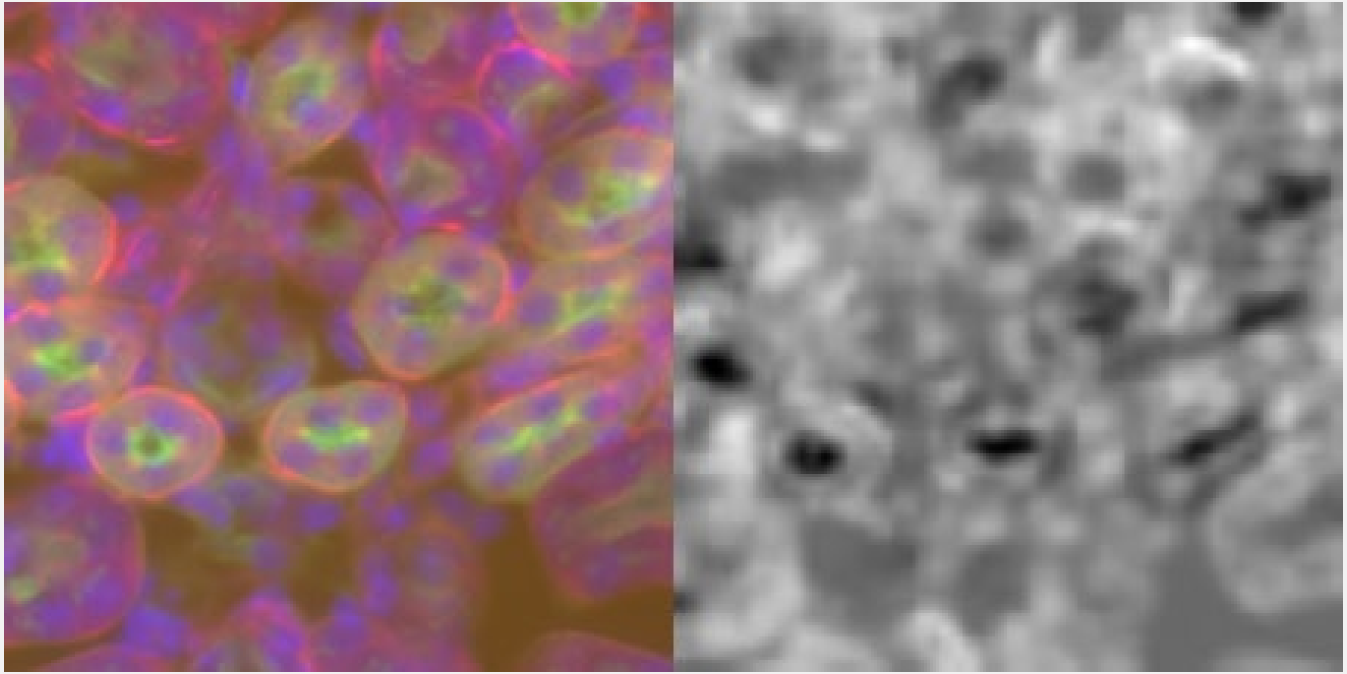
Plot the strongest activation

```
[maxValue,maxValueIndex] = max(max(max(act1)));
act1chMax = act1(:,:,:,maxValueIndex);
act1chMax = mat2gray(act1chMax);
act1chMax = imresize(act1chMax,[256 256]);

I = imtile({Irgb,act1chMax});
```

Plot the strongest activation



Plot the strongest activation



Confusion Matrix

- A confusion matrix shows the predicted vs expected classifications from the trained network

True Class \ Predicted Class	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	923	4	21	8	4	1	5	5	23	6
automobile	5	972	2					1	5	15
bird	26	2	892	30	13	8	17	5	4	3
cat	12	4	32	826	24	48	30	12	5	7
deer	5	1	28	24	898	13	14	14	2	1
dog	7	2	28	111	18	801	13	17		3
frog	5		16	27	3	4	943	1	1	
horse	9	1	14	13	22	17	3	915	2	4
ship	37	10	4	4		1	2	1	931	10
truck	20	39	3	3			2	1	9	923

The blue cells are the correct labels

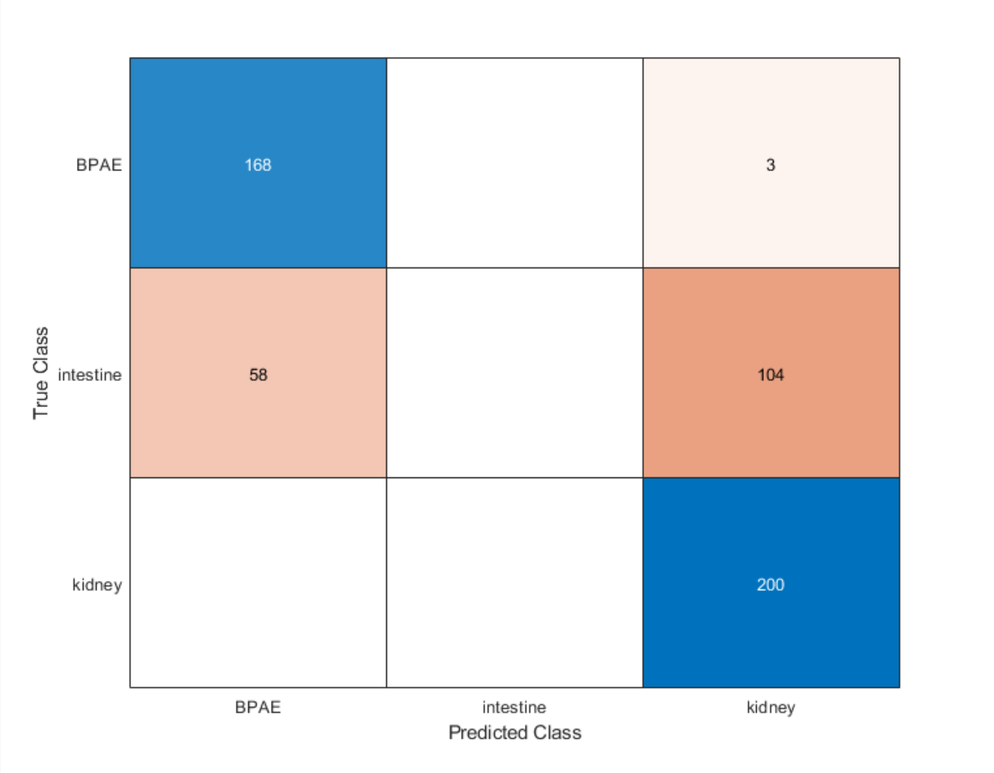
How to visualize the confusion matrix

- Read in the validation images into an Image Datastore
- Use the `classify` method to generate the predicted labels
- Use the function `confusionchart` to display the matrix

```
confusionchart(trueLabels, predictedLabels)
```

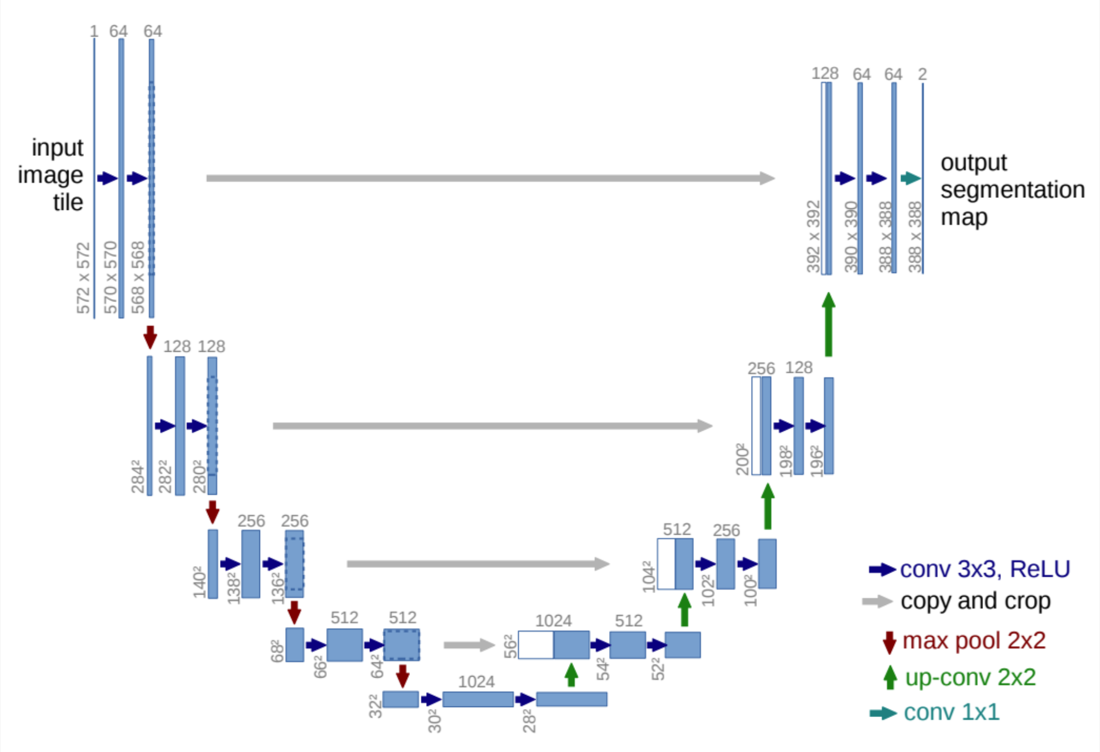
Try this

- Let's try this using either your own network or using the one I supplied


BPAE
intestine
kidney
True Class
168
3
58
104
200
BPAE
intestine
kidney
Predicted Class

Semantic segmentation

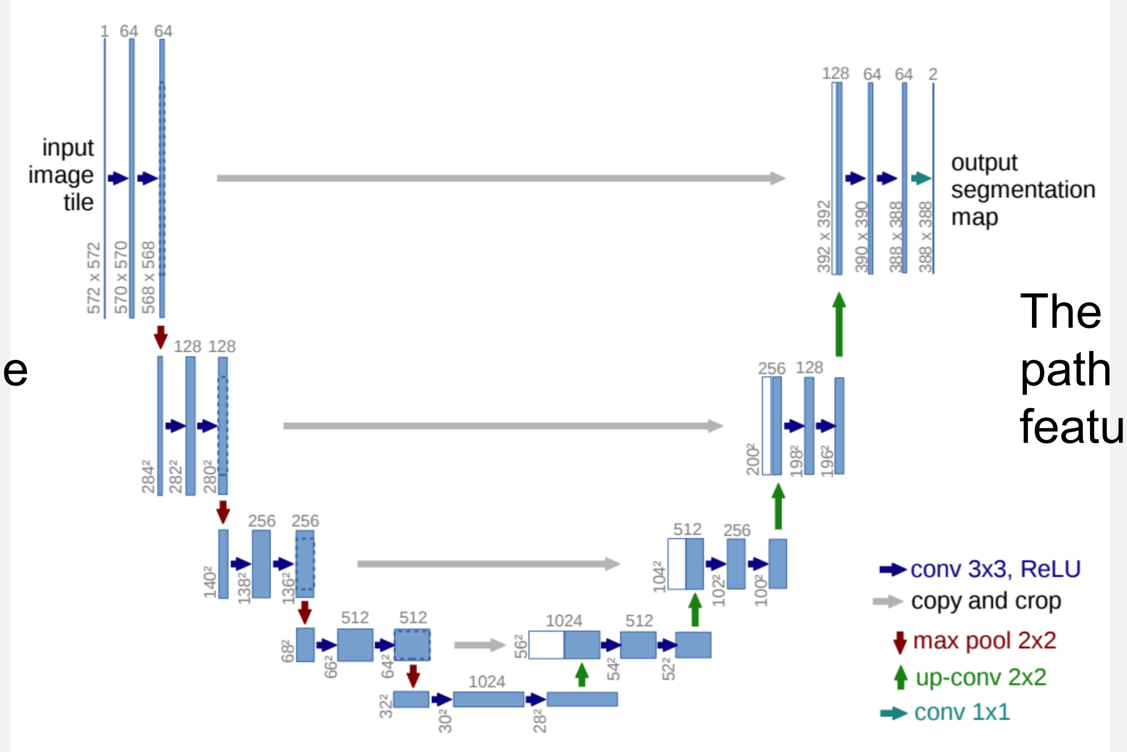
Semantic segmentation using U-net



https://arxiv.org/pdf/1505.04597.pdf

Semantic segmentation using U-net

The downward path learns image features



The upward path marks where features are

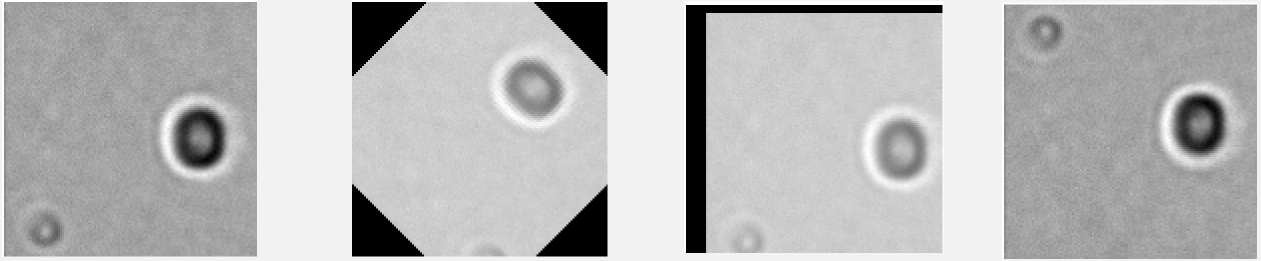
https://arxiv.org/pdf/1505.04597.pdf

Setting up the Unet in MATLAB

```
lgraph = unetLayers(imageSize, numClasses)
```

Image augmentation

- To successfully train the U-net, you need 10k – 100k images
- But we clearly do not have that
- One way to get more images to use "image augmentation":
    - Rotate, translate, reflect images that we have



Test semantic segmentation

```
I = readimage(imds, 10);
C = semanticseg(I, net);

B = labeloverlay(I, C);
imshow(B)
```

Evaluating the results

```
pxdsResults = semanticseg(imds, net);
metrics = evaluateSemanticSegmentation(…
  pxdsResults, pxdsTruth)
```