Introduction to Deep Convolutional Neural Networks

Ye Li (yli192@jhu.edu)
[1]Dept. of Electrical and Computer Engineering, Whiting School of Engineering
[2]Division of Medical Imaging Physics, Dept. of Radiology, School of Medicine
Johns Hopkins University

Context

- Introduction to Neural Networks
- Introduction to Deep Convolutional Neural Networks (DCNN)
- Deep Learning in Medical Image Segmentation
- DCNN Layer Functionality
- DCNN Architecture functionality

Conventional Neural Networks

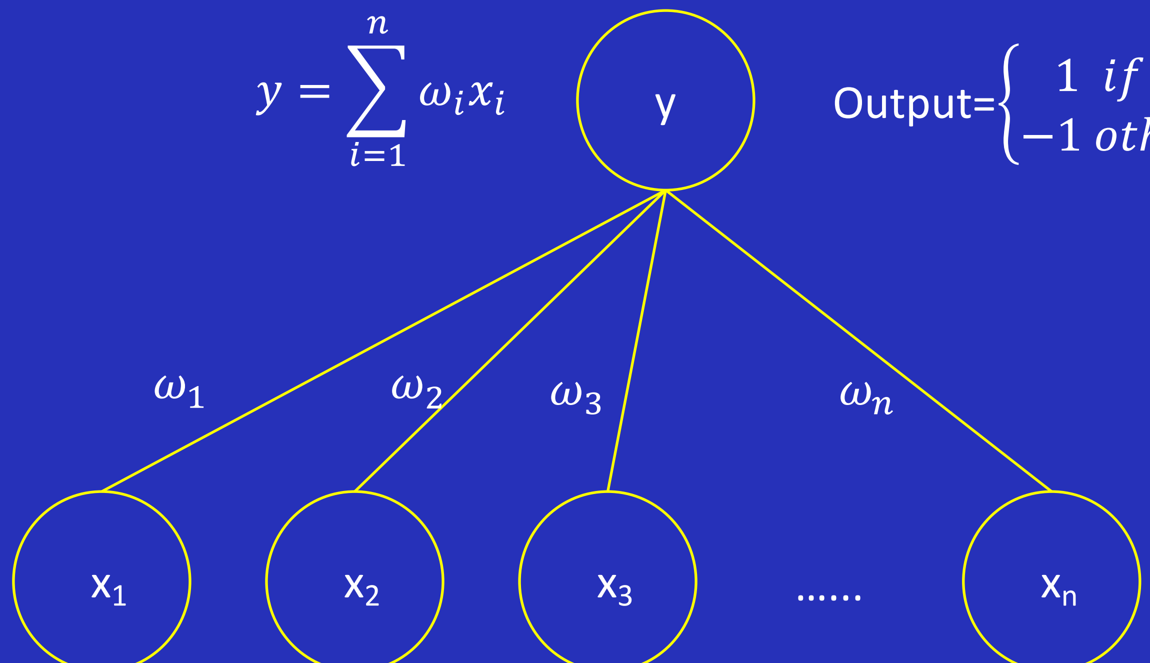
Single-layer Perceptrons (SLP)

- Can classify linearly separable data into binary classes: -1 and 1.
- A feed-forward network based on a threshold transfer function

Output layer

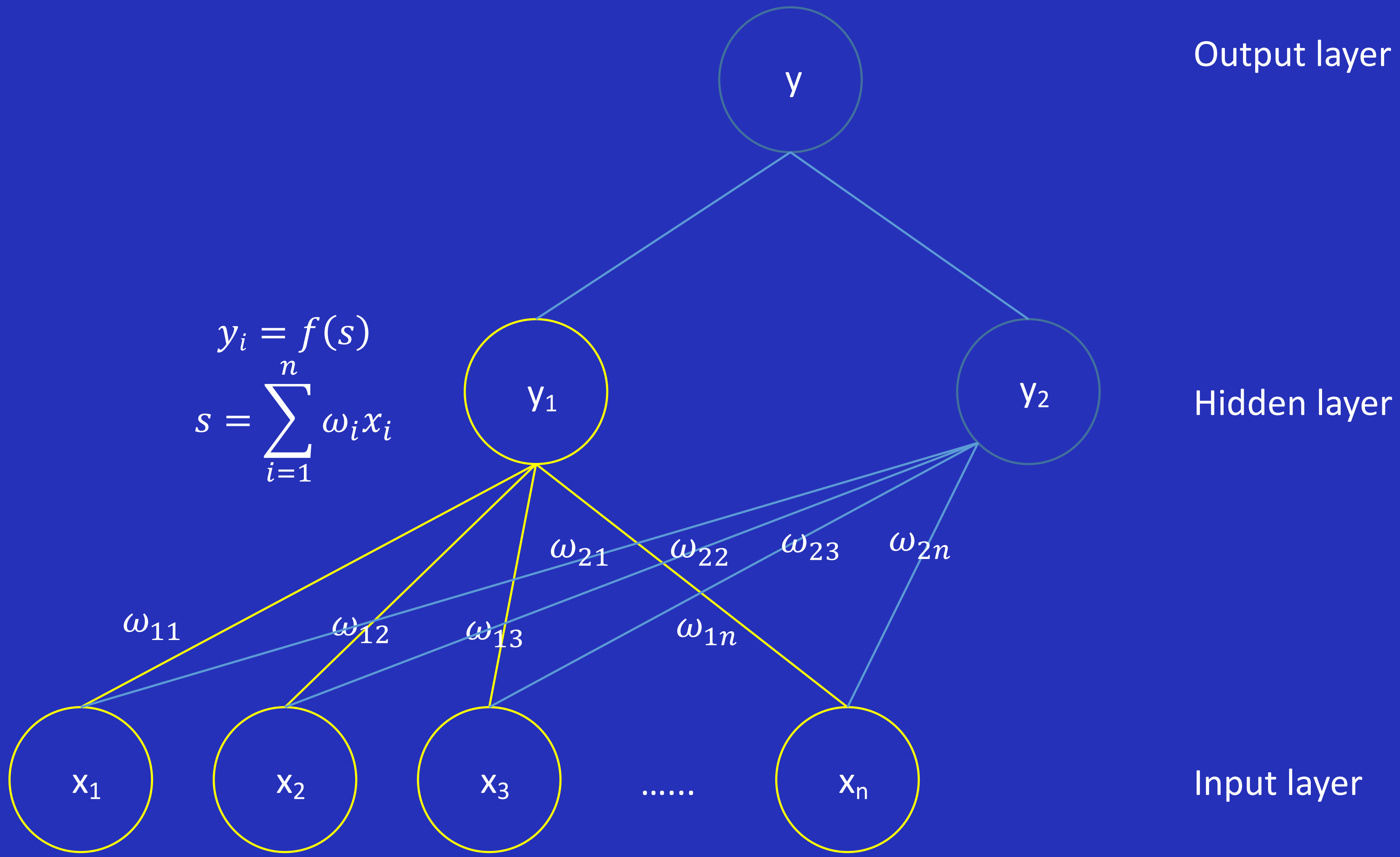
$$y = \sum_{i=1}^{n} \omega_i x_i$$

$$\text{Output=}\begin{cases} 1 \ \ if\ y > \theta \\ -1\ otherwise \end{cases}$$



Input layer

Multi-layer Perceptrons (MLP)



About MLP

- Differs from SLP by two things:
  - A soft thresholding function after each summation
  - Introduction of hidden layers
- Many levels can be specified to model non-linear relationship
- The number of hidden units is related to the capacity of the perceptron

Backpropagation

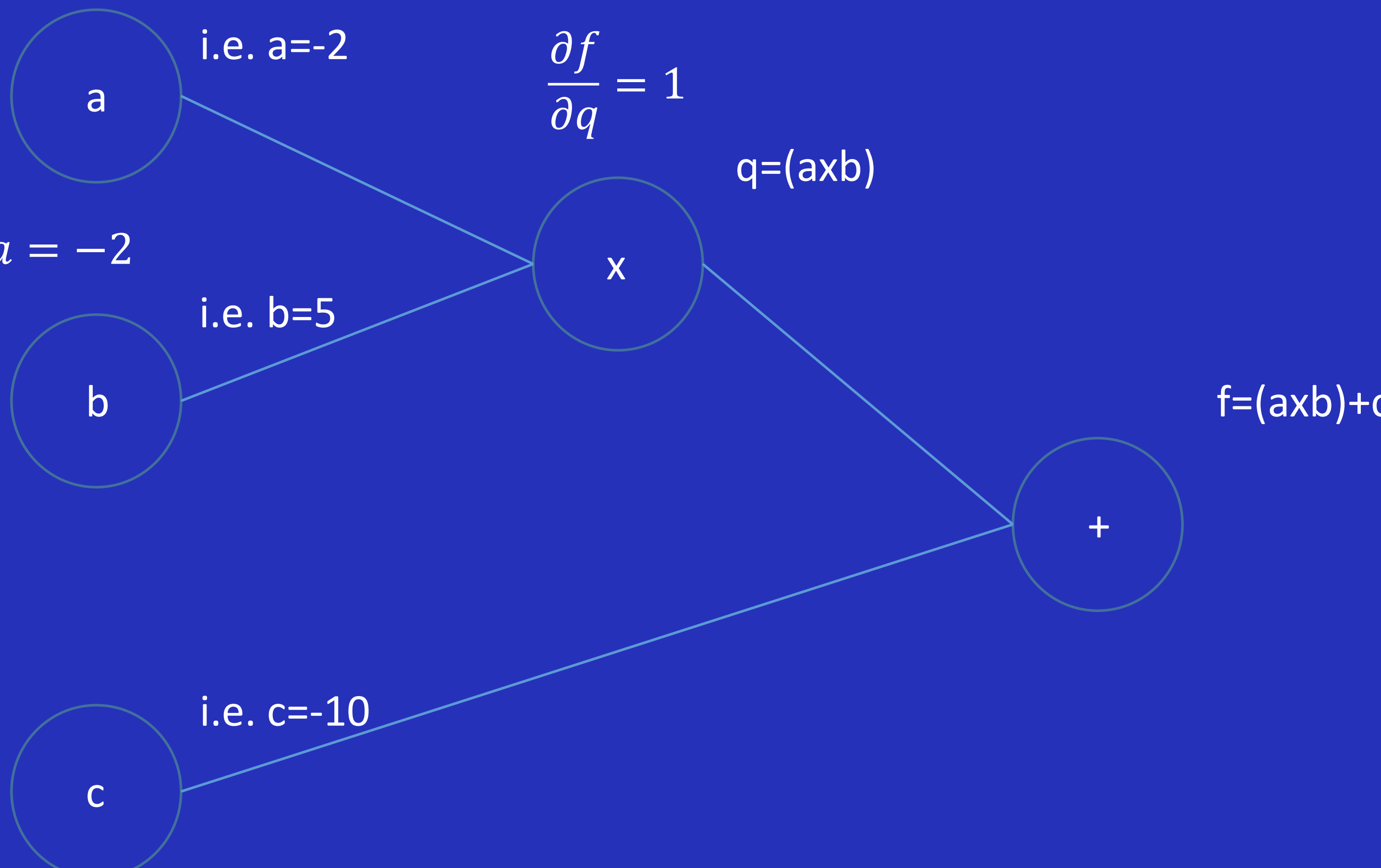
- To apply the chain rule many many times to calculate the gradient of a loss function with respect to all the inputs (weights, input data) in the network.

Backpropagation (continued)

$$\frac{\partial f}{\partial a} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial a} = b = 5$$

$$\frac{\partial f}{\partial b} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial b} = a = -2$$

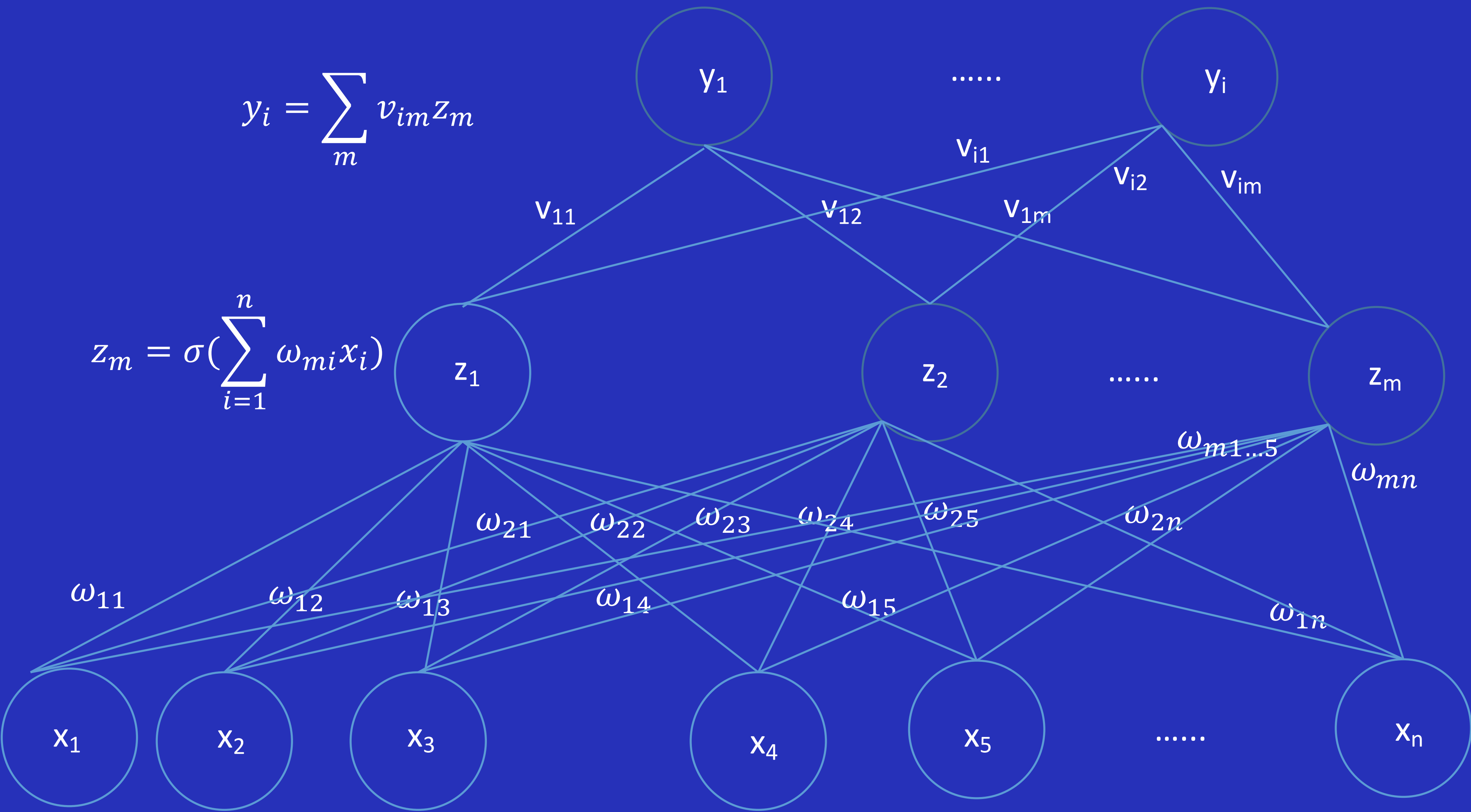
$$\frac{\partial f}{\partial c} = 1$$



Backpropagation for MLP

$$y_i = \sum_m v_{im} z_m$$

$$z_m = \sigma\left(\sum_{i=1}^{n} \omega_{mi} x_i\right)$$



Backpropagation for MLP (cont'd)

- Loss function

$$E[\omega,\nu] = \sum_i \{y_i - \sum_m \nu_{im}\sigma(\sum_n \omega_{mn}x_n)\}^2$$

- Update terms are negative derivatives of the loss w.r.t the local parameters (weights)

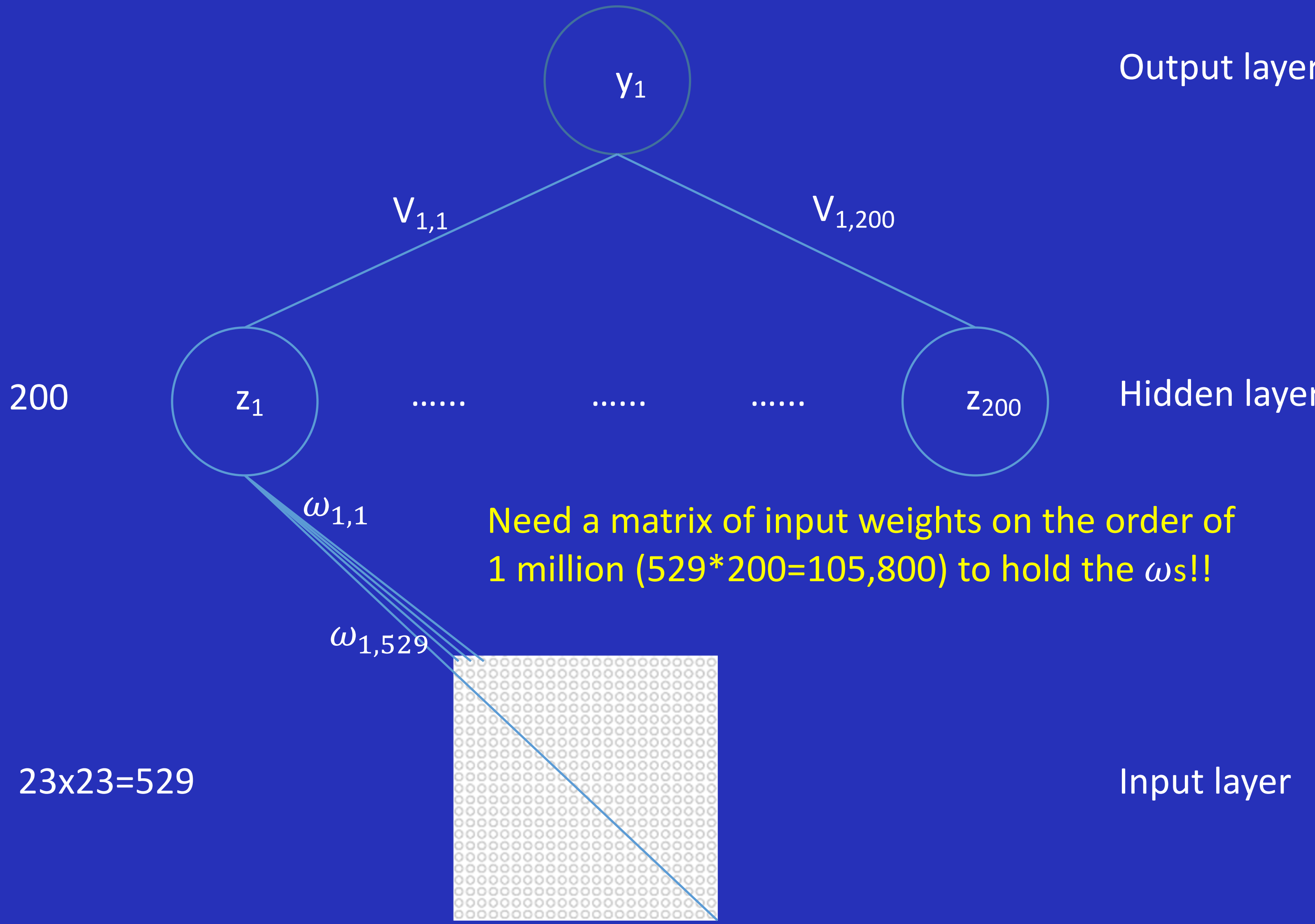
$$\Delta\omega_{mn} = -\frac{\partial E}{\partial \omega_{mn}} \qquad \Delta\nu_{im} = -\frac{\partial E}{\partial \nu_{im}}$$

- By defining $z_m = \sigma(\sum_{i=1}^{n} \omega_{mi}x_i)$ and $E = \sum_{i=1}^{n}(y_i - \sum_m v_{im}z_m)^2$ … … …

$$\frac{\partial E}{\partial \omega_{mn}} = 2\sum_i (y_i - \sum_m (\nu_{im}z_m))\nu_{im}x_n\sigma(\sum_n \omega_{mn}x_n)\{1 - \sigma(\sum_n \omega_{mn}x_n)\}$$

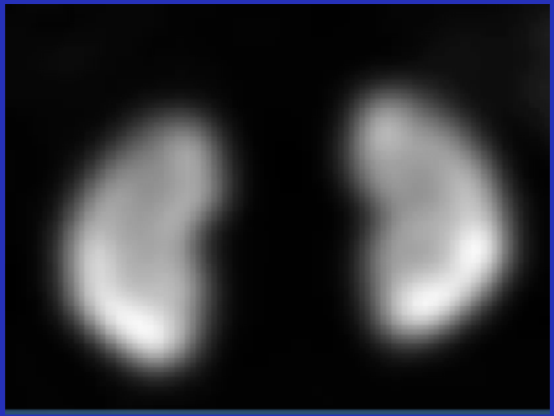
Detailed derivations are available at:
http://garyliye.com/Multilayer_perceptron_and_backpropagation.pdf

What about a Real-world Image?

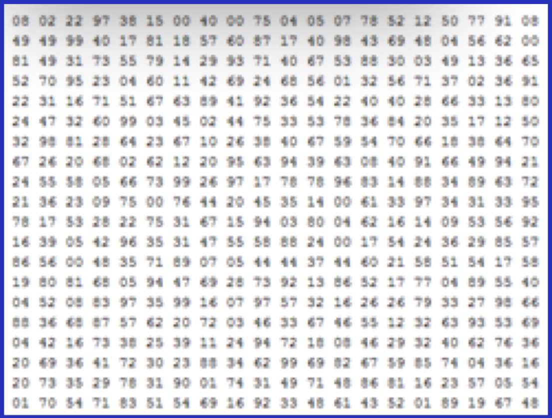


Need a matrix of input weights on the order of 1 million (529*200=105,800) to hold the ωs!!

Spatial Structure



What we see



08	02	22	97	38	15	00	40	00	75	04	05	07	78	52	12	50	77	91	08
49	49	99	40	17	81	18	57	60	87	17	40	98	43	69	48	04	56	62	00
81	49	31	73	55	79	14	29	93	71	40	67	53	88	30	03	49	13	36	65
52	70	95	23	04	60	11	42	69	24	68	56	01	32	56	71	37	02	36	91
22	31	16	71	51	67	63	89	41	92	36	54	22	40	40	28	66	33	13	80
24	47	32	60	99	03	45	02	44	75	33	53	78	36	84	20	35	17	12	50
32	98	81	28	64	23	67	10	26	38	40	67	59	54	70	66	18	38	64	70
67	26	20	68	02	62	12	20	95	63	94	39	63	08	40	91	66	49	94	21
24	55	58	05	66	73	99	26	97	17	78	78	96	83	14	88	34	89	63	72
21	36	23	09	75	00	76	44	20	45	35	14	00	61	33	97	34	31	33	95
78	17	53	28	22	75	31	67	15	94	03	80	04	62	16	14	09	53	56	92
16	39	05	42	96	35	31	47	55	58	88	24	00	17	54	24	36	29	85	57
86	56	00	48	35	71	89	07	05	44	44	37	44	60	21	58	51	54	17	58
19	80	81	68	05	94	47	69	28	73	92	13	86	52	17	77	04	89	55	40
04	52	08	83	97	35	99	16	07	97	57	32	16	26	26	79	33	27	98	66
88	36	68	87	57	62	20	72	03	46	33	67	46	55	12	32	63	93	53	69
04	42	16	73	38	25	39	11	24	94	72	18	08	46	29	32	40	62	76	36
20	69	36	41	72	30	23	88	34	62	99	69	82	67	59	85	74	04	36	16
20	73	35	29	78	31	90	01	74	31	49	71	48	86	81	16	23	57	05	54
01	70	54	71	83	51	54	69	16	92	33	48	61	43	52	01	89	19	67	48

We computers see

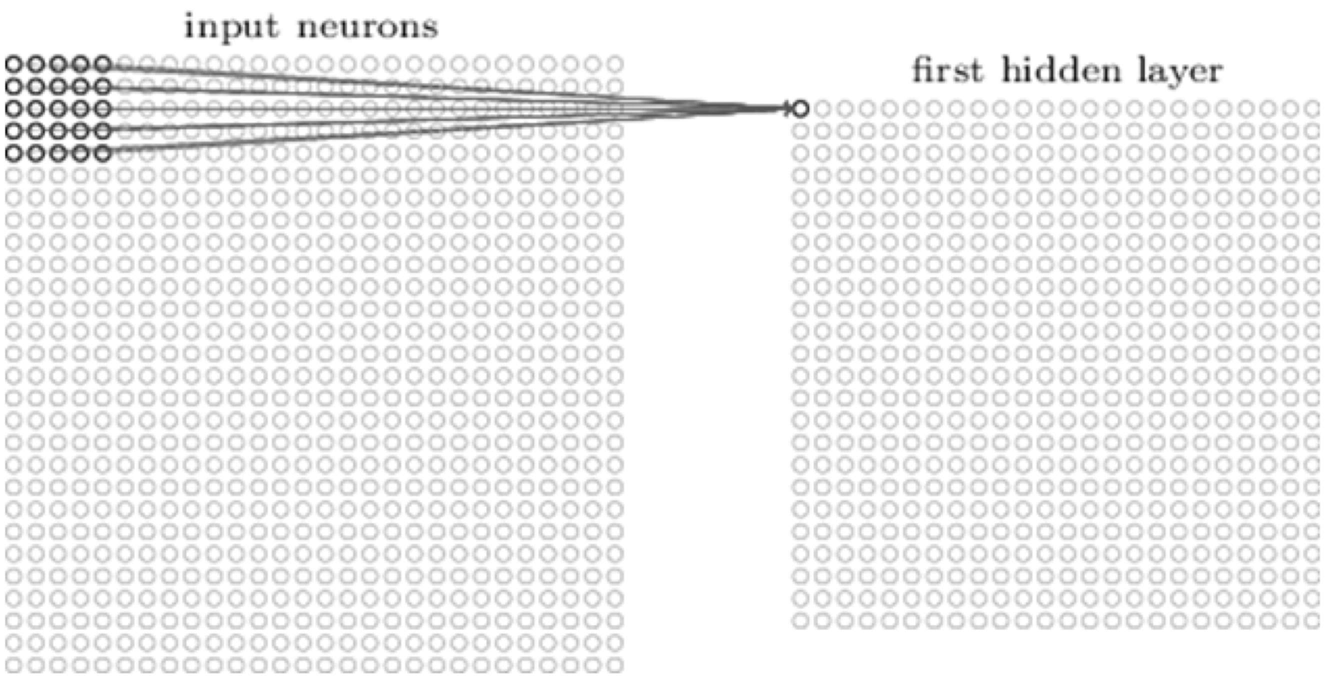
Vectorizing an image completely ignores the complex 2D spatial structure of an image

Limitations of Conventioanl Neural Networks

- Impractical for real-world image classification
- Ignores 2D/3D spatial structure in image
- Solution to overcome both these disadvantages?

One solution: Convolution

Use 2D convolution instead of matrix multiplications:
-Learning a set of convolutional filters (each of 5x5,say) is much more tractable than learning a large matrix (529x200)



Visualization of 5 x 5 filter convolving around an input volume and producing an activation map

https://adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks/

Convolutional Neural Networks

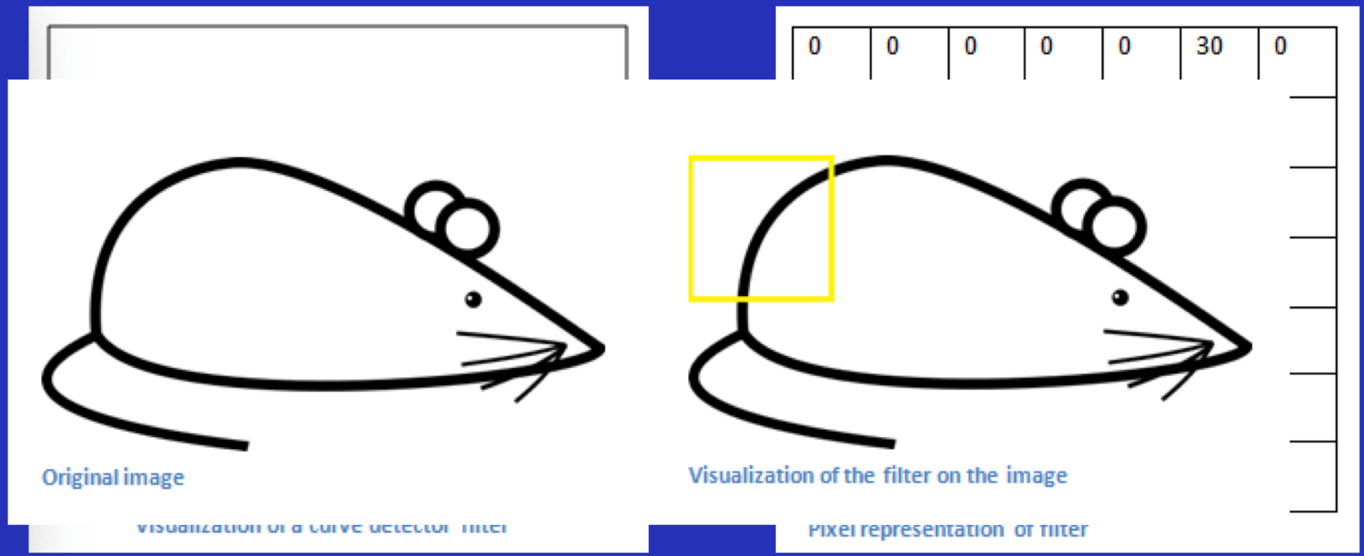
Convolutional Neural Networks (CNN)

- CNN has proven very powerful
  -Retains structural or configural information in neighboring pixels or voxels in a (medical) image
  -Exploits extensive weight-sharing to reduce the degrees of freedom of models
  -Composed of convolution layers interspersed with pooling (sub-sampling) layers
  -Highly parallelizable
  -GPU implementations can accelerate 40 times or more
  -Trained using backpropagation algorithm and lots of labeled data
  -First uses in medical imaging in 1990's
- Improvement of artificial neural networks
  - More layers, higher levels of abstraction, improved predictions

H. Greenspan et al., IEEE TMI, May 2016

High Level Perspective of Convolution

- Convolutional filters are essentially feature identifiers
- Features can be high-level (abstract) and low-level such as straight edges, simple colors, and curves.



https://adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks/

High Level Perspective of Convolution (cont’d)

The output of the filter has a high activation value. Or say, the neuron is fired/excited!

High activation

Visualization of the receptive field

0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

Pixel representation of the receptive field

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

Multiplication and Summation = (50*30)+(50*30)+(50*30)+(20*30)+(50*30) = 6600 (A large number!)

The output of the filter has a low activation value.

No activation

Visualization of the filter on the image

0	0	0	0	0	0	0
0	40	0	0	0	0	0
40	0	40	0	0	0	0
40	20	0	0	0	0	0
0	50	0	0	0	0	0
0	0	50	0	0	0	0
25	25	0	50	0	0	0

Pixel representation of receptive field

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

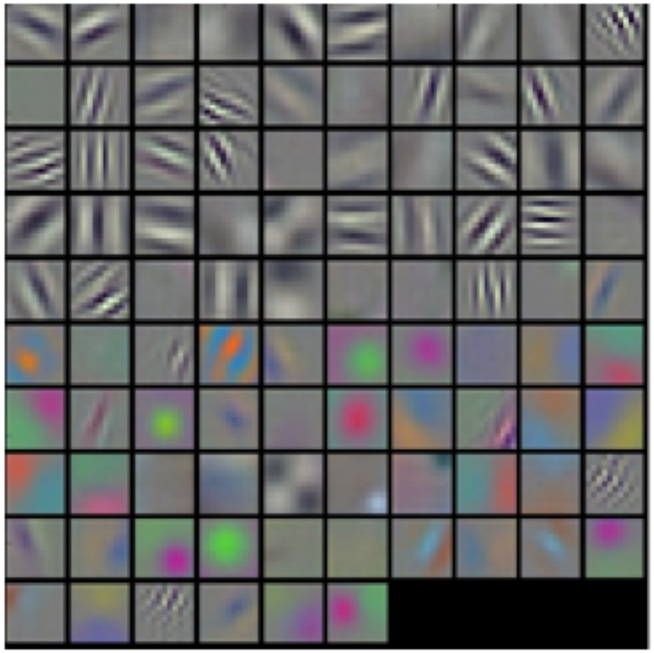
Multiplication and Summation = 0

https://adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks/

1^{st} Conv Layer **Filters** Learned in AlexNet

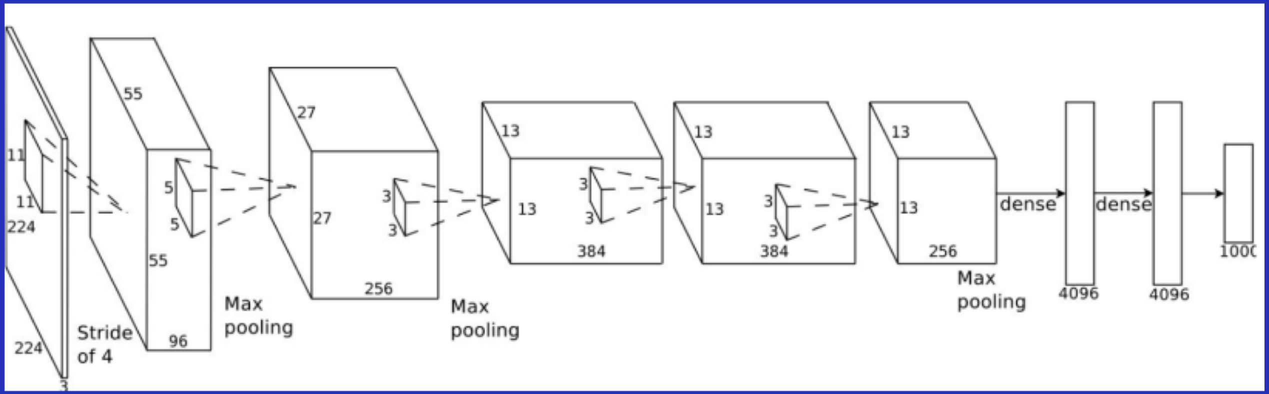
Example filters learned by Krizhevsky et al. Each of the 96 filters shown below is of size 11x11x3.

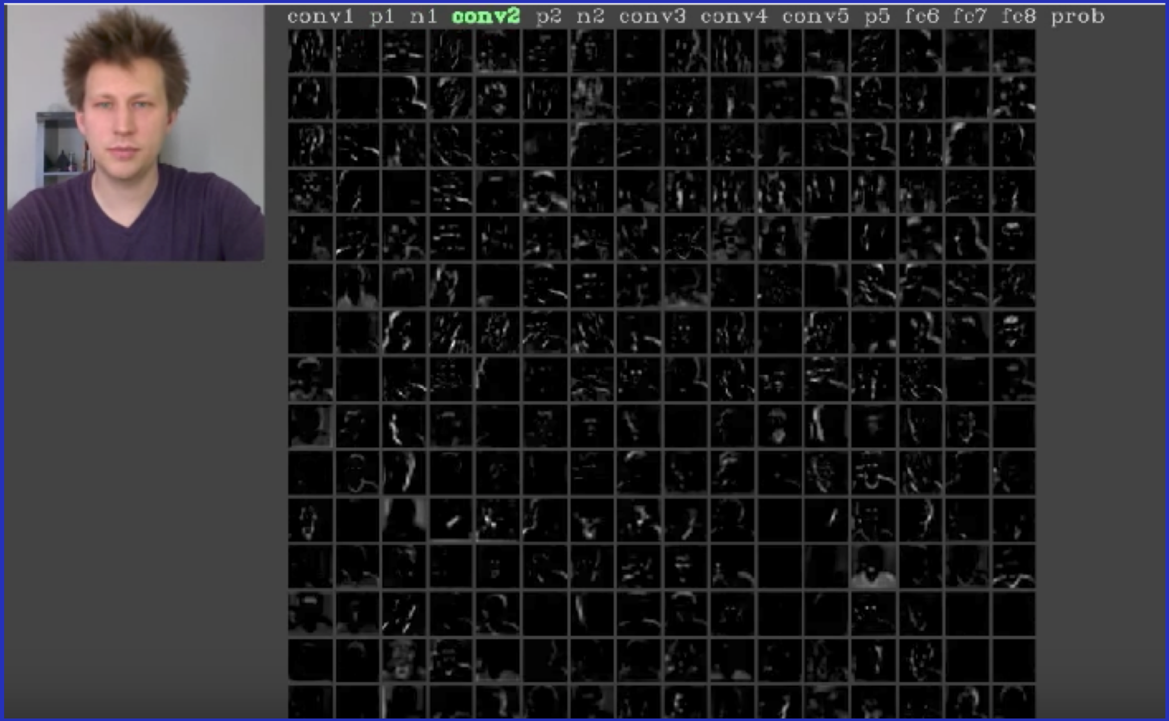
Each layer of the activation map(s) is basically describing the locations in the original image for where certain low level features appear.



Visualizations of filters

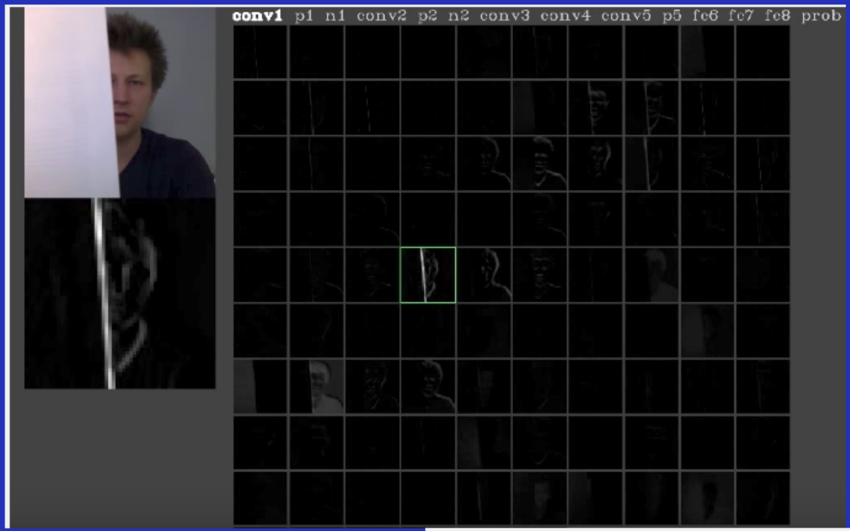
The 2nd Conv Layer Activation Map

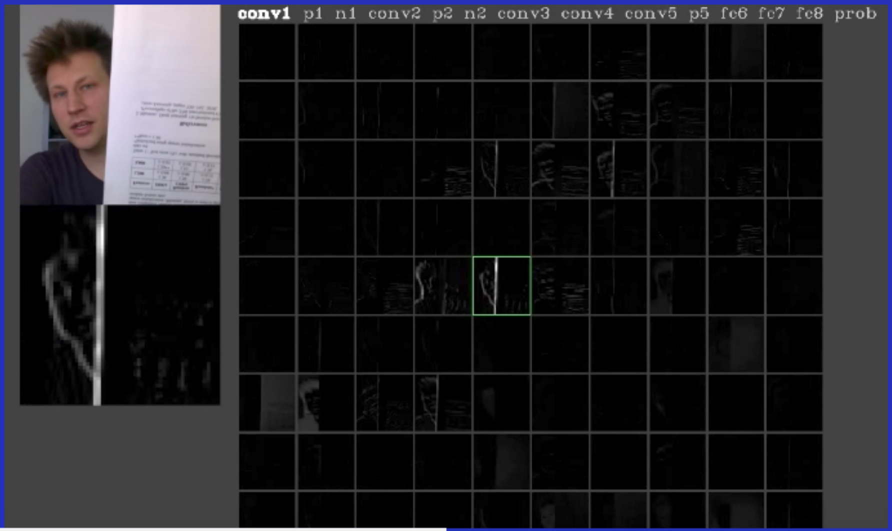


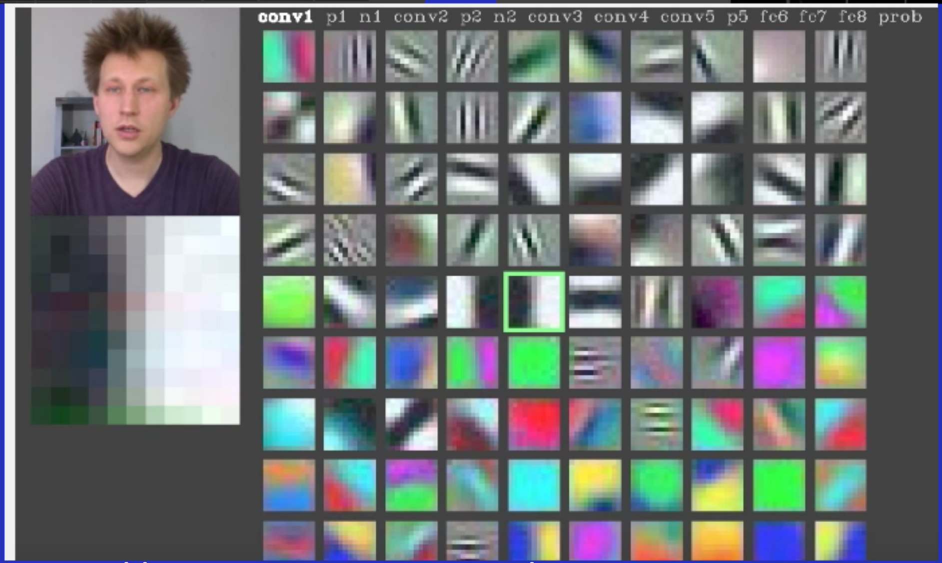


https://www.youtube.com/watch?v=AgkfIQ4IGaM

Filters and Activation Maps







https://www.youtube.com/watch?v=AgkfIQ4IGaM

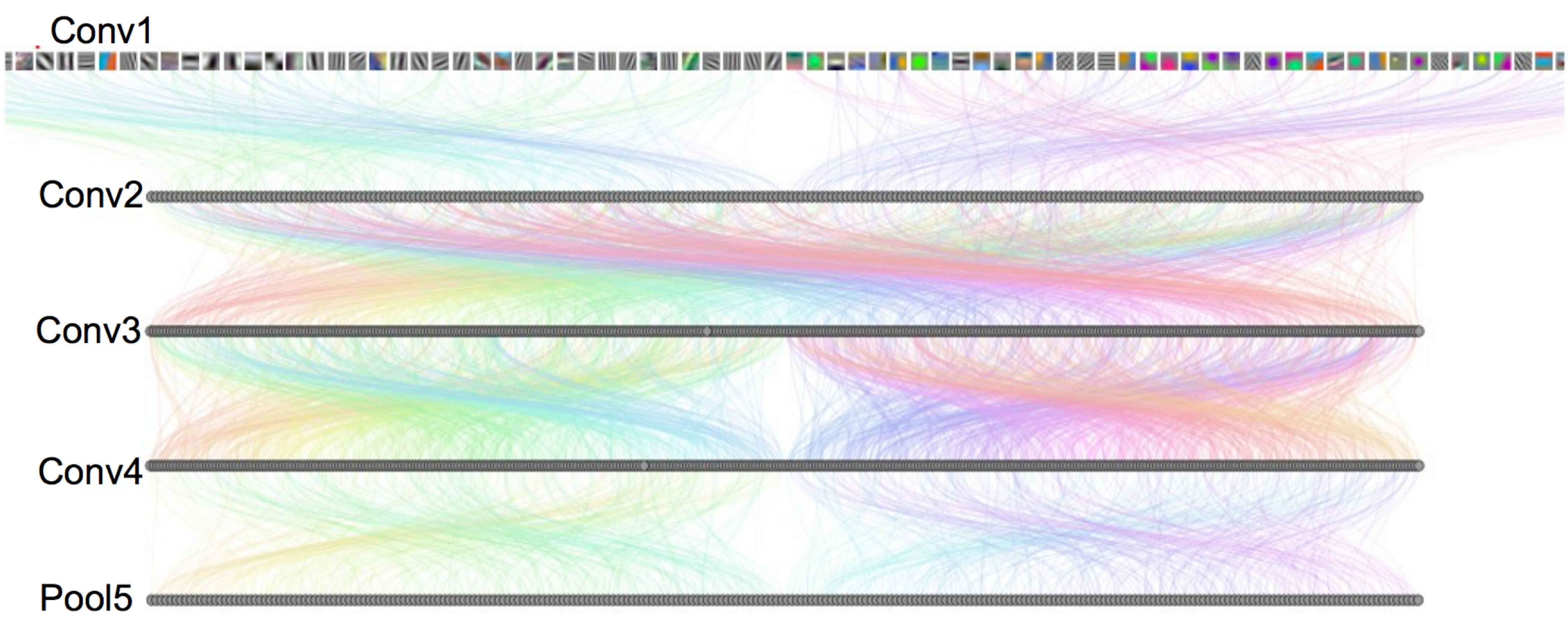
Connection Weights Between Convolutional Layers

- Let the learnable connection weights connecting feature map i at layer $l-1$ and the feature map j at the layer l be k_{ij}^{l}. Specifically, the units of the convolutional layer l compute their activations A_j^l based only on a spatially contiguous subset of units in the feature maps A_i^{l-1} of the preceding layer $l-1$ by convolving the kernels k_{ij}^{l} as follows:

$$A_j^l = f\left(\sum_{i=1}^{M(l-1)} A_i^{l-1} * k_{ij}^{l} + b_j^l\right)$$

Say if there are 5 feature maps at layer $l-1$ and 4 feature maps at layer l, there would be 4 axax5(depth of the feature map at previous layer) connection weights

How objects are represented in CNN?



Bolei Zhou, et al., arXiv.org

Neuroscience connection

- Similar (convolution-like) computations within the human brain
- Primary visual cortex has simple and complex cells
- The simple cells responded primarily to oriented edges and gratings
- The complex cells were also sensitive to these edges and grating but exhibited spatial invariance

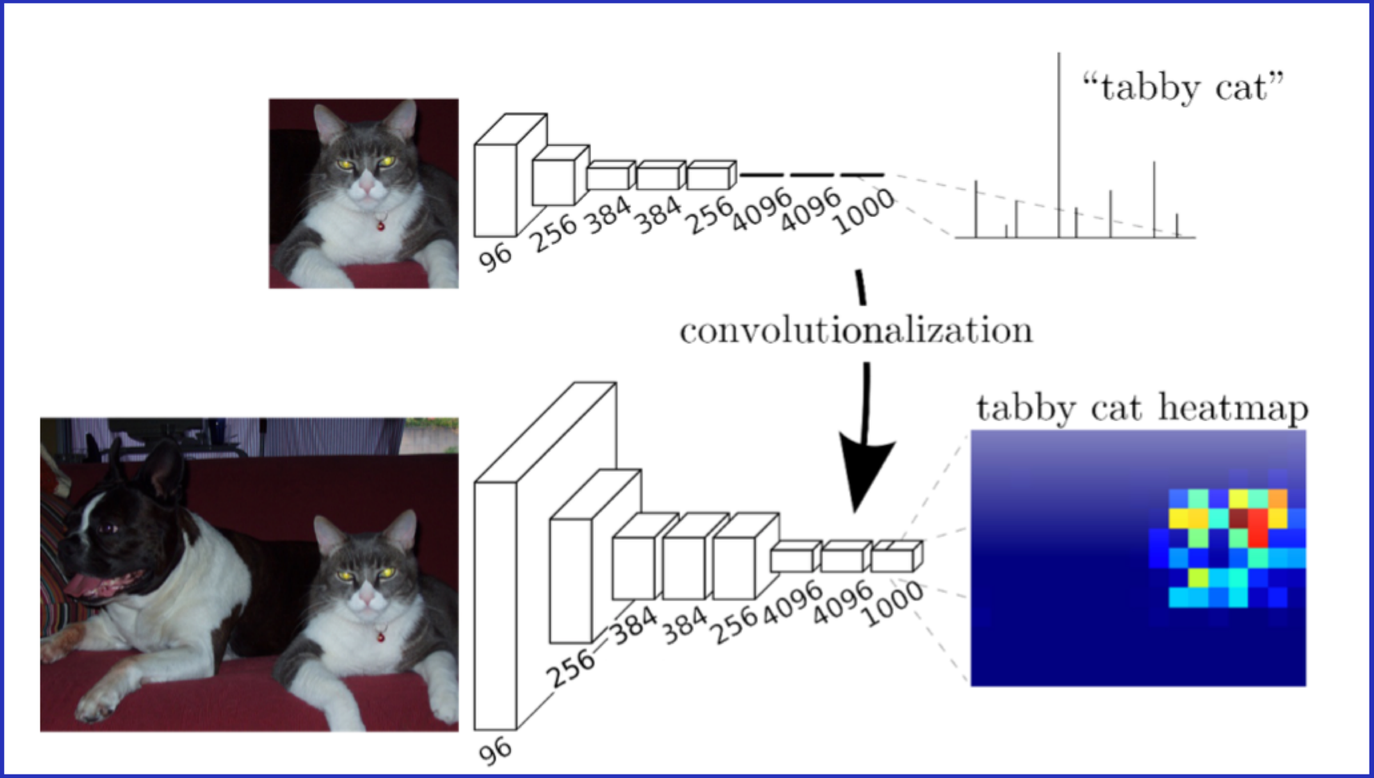
Deep Learning in Medical Imaging

- Difficult to obtain large enough training data
- Some solutions to lack of “big data” in medical imaging
- What architecture to use?

H. Greenspan et al., IEEE TMI, May 2016

Segmentation: pixel-wise classification

Transforming fully connected layers into convolution layers enables a classification net to output a spatial map.



Shelhamer, E. Long, J. Darrell, T. IEEE Trans Pattern Anal Mach Intell, 2016

Network Depth and Receptive Field Size

- As you go deeper into the network, the filters begin to have a larger and larger receptive field, which means that they are able to consider information from a larger area of the original input volume (another way of putting it is that they are more responsive to a larger region of pixel space)

Layer functionality

Convolutional layer

- Local connectivity
  -Because we use convolutional filter with size much smaller than the image it operates on. This contrasts with the global connectivity paradigm relevant to vectorized images
- Weight sharing
  -The same filter applied across the image
- Can be seen as a **local independent feature-detector;** To detect local features (local connectivity) at different position in the input feature maps

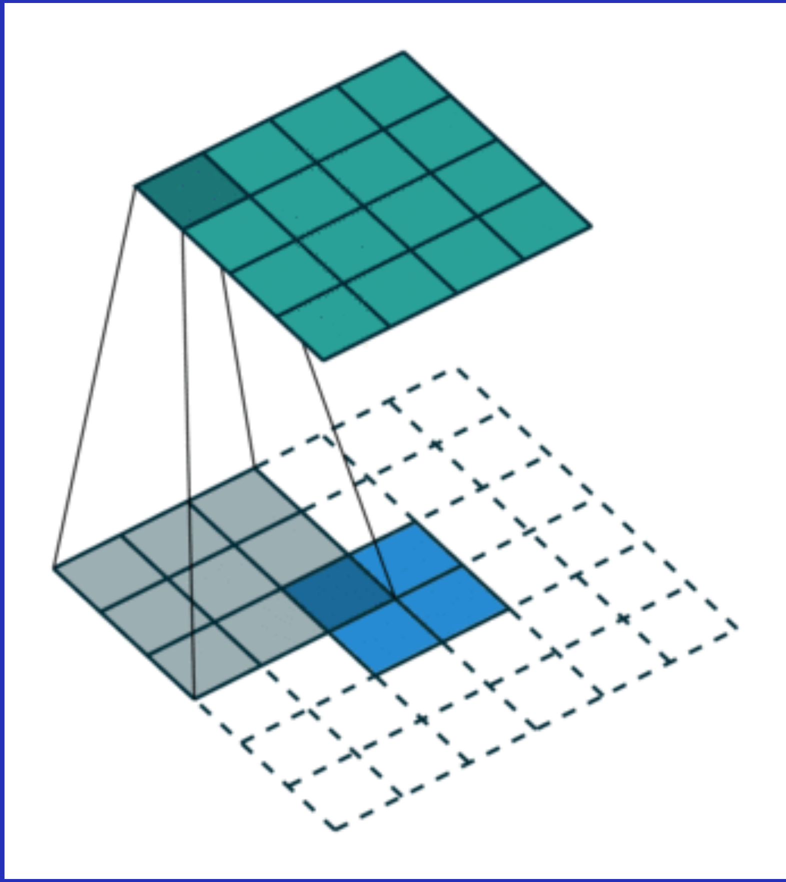
Max-Pooling

- The Neocognitron model inspired the modeling of simple cells as convolutions.
- The complex cells can be modeled as a max-pooling operation, which can be thought as a max filter.
- Picks the highest activation in a local region, thus providing a small degree of spatial invariance, which is analogous to the operation of complex cells.

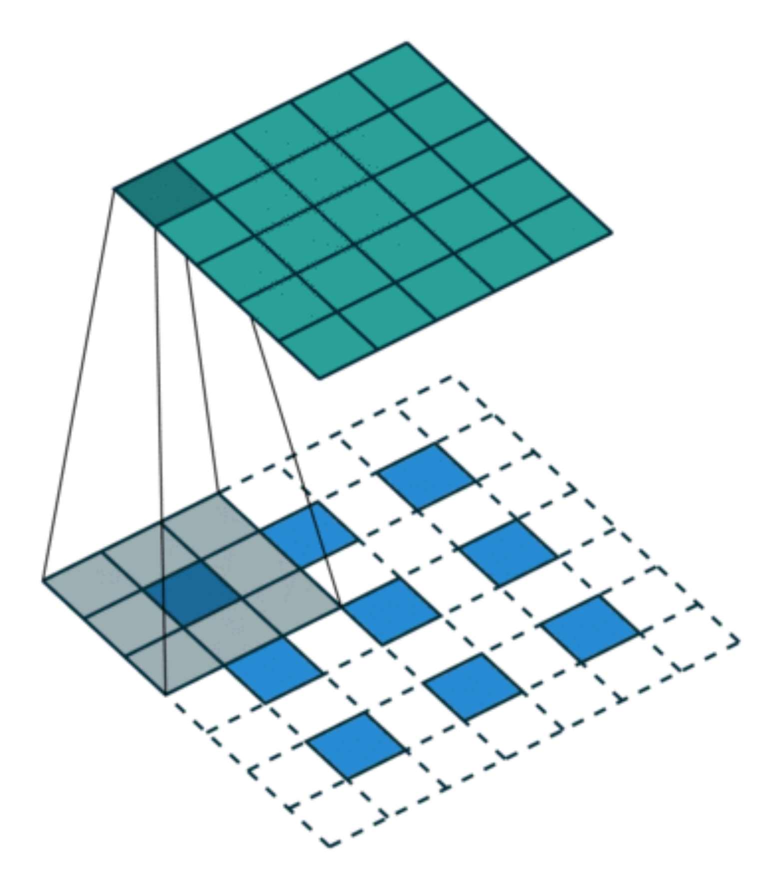
Non-linearity layer

- Necessary because cascading linear (like convolution) systems is another linear system
- Non-linearity between layers ensure that the model is more expressive than a linear model
- In theory, no non-linearity has more expressive power than any other, as long as they are continuous, bounded, and monotonically increasing.
- Maas et al. introduced a new kind of nonlinearity, called the leaky-ReLU. ReLU(x)=max(0,x)+bmin(0,x)

Deconvolutional Layer



Without Padding

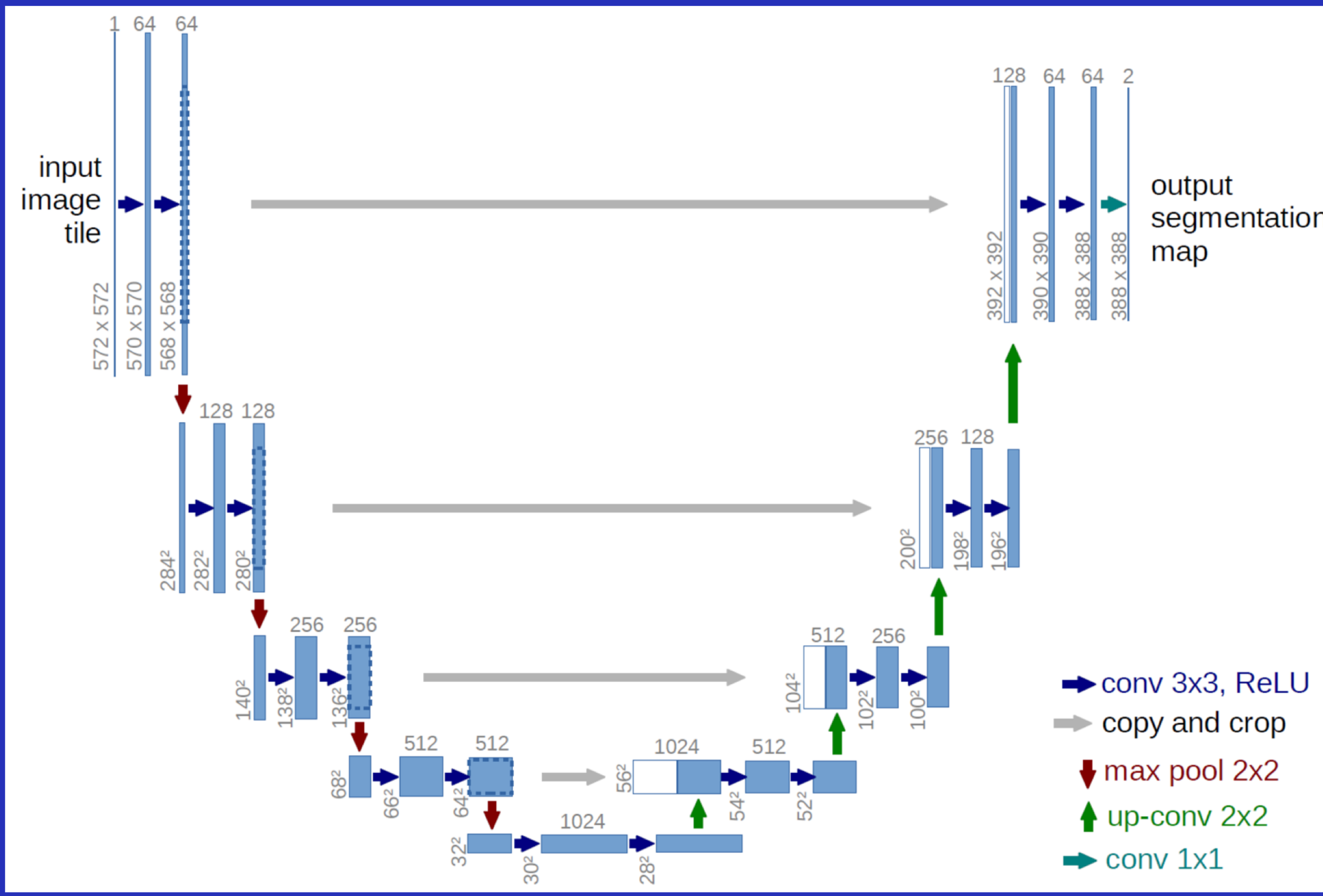


With Padding

https://datascience.stackexchange.com/questions/6107/what-are-deconvolutional-layers

Architecture functionality (segmentation)

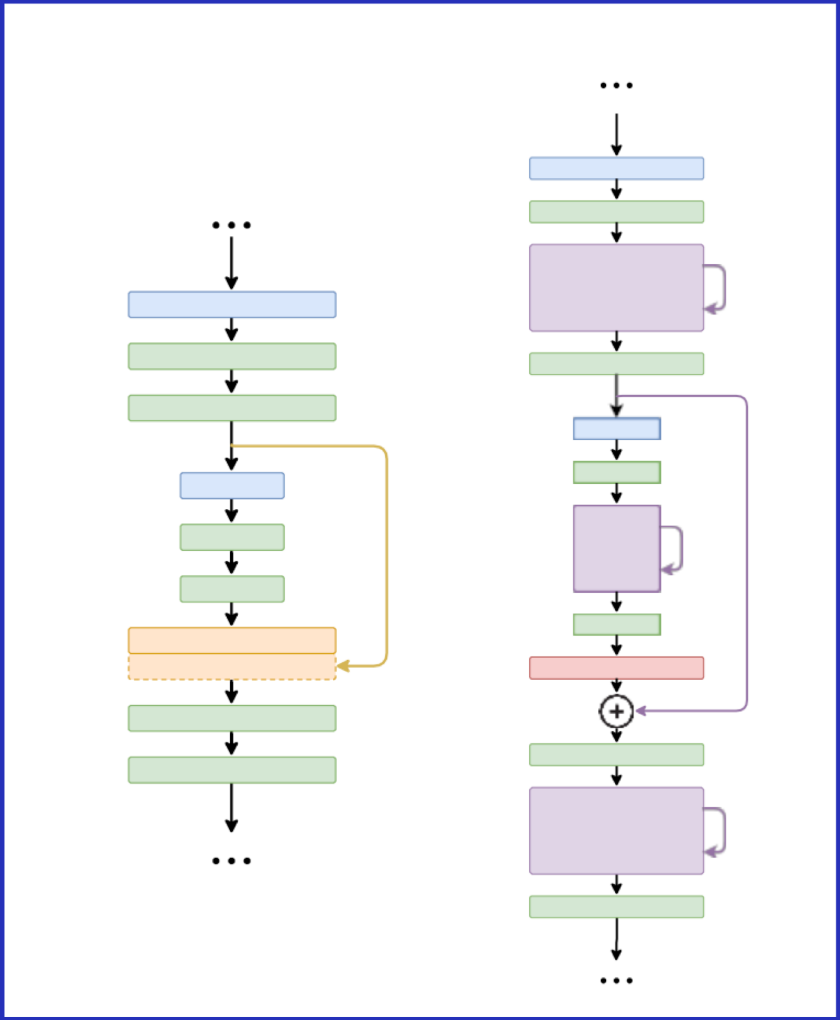
Encoder-decoder architecture (U-net)



Olaf R. et al., **arXiv.org**

Summation based skip architecture

U-net:
Copy
and past



FusionNet:
Copy and
add

Tran M.Q, et al., **arXiv.org**