MIDDLE EAST TECHNICAL UNIVERSITY

# DI504 Foundations of Deep Learning

Convolutional Neural Networks

## Welcome again!

This is DI504, Foundations of Deep Learning

- We previously talked about feature spaces and score functions, which are basically the fundamentals concept of machine learning.
- Then we jumped into loss function. We learned what ANNs are and how to optimized them.
- Now, we will talk about a special type of ANNs, namely the convolutional neural networks (CNNs).

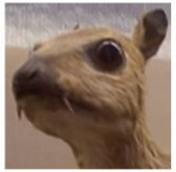
## Convolution

- Mathematically speaking, the convolution of f and g (written as f*g), is defined as the integral of the product of the two functions after one is reversed and shifted (assuming f and g are both real):

$$(f * g) = (g * f) = \sum_{m=-M}^{M} f[m] \cdot g[n-m]$$

## Convolution (in signal processing)

- In signal (image, audio, speech, etc.) processing, convolutions of signals are practically carried out by so-called “filters” or “kernels”.
- In this notation, the size of the filter/kernel represents the limits of the summation in the above equation.
- Convolution of an image with different filters can perform operations such as edge detection, blur and sharpen by applying filters, just to name a few.



$$* \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} + 0 =$$

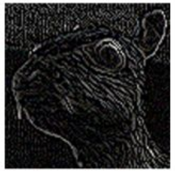
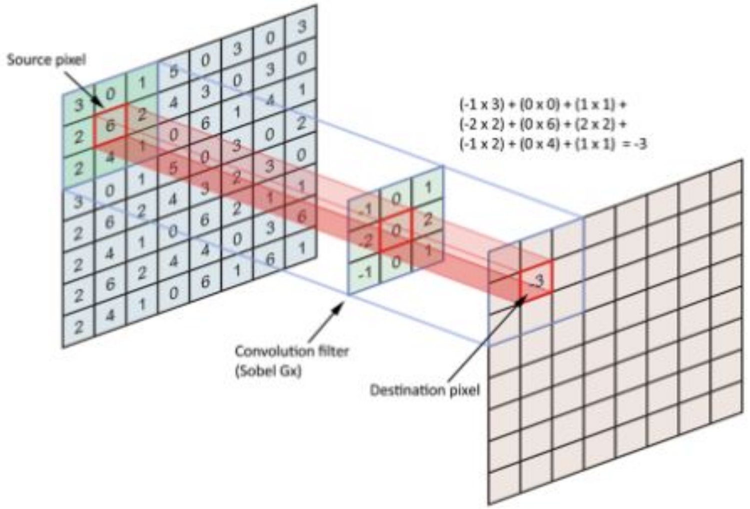


*Figure 1: An example of Laplacian edge detection effect, achievable by convolving a kernel and an image.*

METU

## Convolution on an Image

- In image processing, because the signal and the kernel are two-dimensional, the summation operator becomes a double summation.
- In this sense, the result of the convolution is the summation of the pixel-wise multiplication of the kernel values, with the neighbouring pixel values, centred around an arbitrary pixel



Source pixel
(-1 x 3) + (0 x 0) + (1 x 1) +
(-2 x 2) + (0 x 6) + (2 x 2) +
(-1 x 2) + (0 x 4) + (1 x 1) = -3
Convolution filter
(Sobel Gx)
Destination pixel

*Figure 2. Convolution is the summation of the pixel-wise multiplication of the kernel values, with the neighbouring pixel values, centred around an arbitrary pixel.*

METU

## Convolution vs Correlation (same?)

In CNN literature, the convolution operation is used interchangeably with another operation, called the “correlation”.

Theoretically, convolution are linear operations on the signal or signal modifiers, whereas correlation is a measure of similarity between two signals.

The basic difference between convolution and correlation is that the convolution process rotates the kernel by 180 degrees, in other words, g (the filter) is reversed. On the other hand, in correlation, it is simply not.

For real-valued functions (which is usually the case for deep CNNs), the convolution of f[m] ***** g[m] is simply the (cross-) correlation of f[m] ★ g[-m] or f[-m] ★ g[m].

$$(f \star g) = (g \star f) = \sum_{m=-M}^{M} f[m] \cdot g[n+m]$$

# Biological Background

- During 1964, David Hubel and Torsten Wiesel studied the short and long term effects of depriving kittens of vision in one eye.
- In their experiments, Wiesel and Hubel used kittens as models for human children.
- Hubel and Wiesel researched whether the impairment of vision in one eye could be repaired or not and whether such impairments would impact vision later on in life.

**Hubel & Wiesel,**

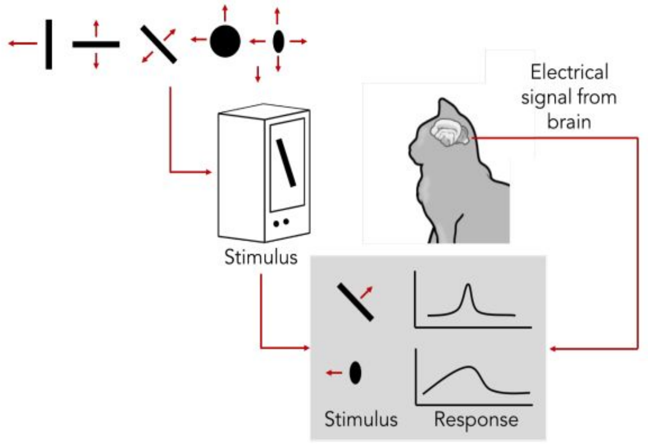
1959

RECEPTIVE FIELDS OF SINGLE NEURONES IN
THE CAT'S STRIATE CORTEX

1962

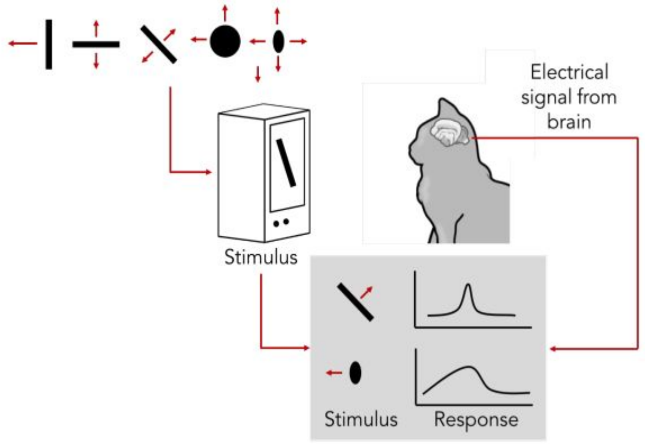
RECEPTIVE FIELDS, BINOCULAR INTERACTION
AND FUNCTIONAL ARCHITECTURE IN
THE CAT'S VISUAL CORTEX

1968...



# Biological Background

- Hubel and Wiesel discovered two major cell types in the primary visual cortex (V1) of cats [7].
- The first type, the simple cells, respond to bars of light or dark when placed at specific spatial locations. Each cell has an orientation of the bar at which it fires most, with its response falling off as the angle of the bar changes from this preferred orientation (creating an orientation 'tuning curve').
- The second type, complex cells, have less strict response profiles; these cells still have preferred orientations but can respond just as strongly to a bar in several different nearby locations.



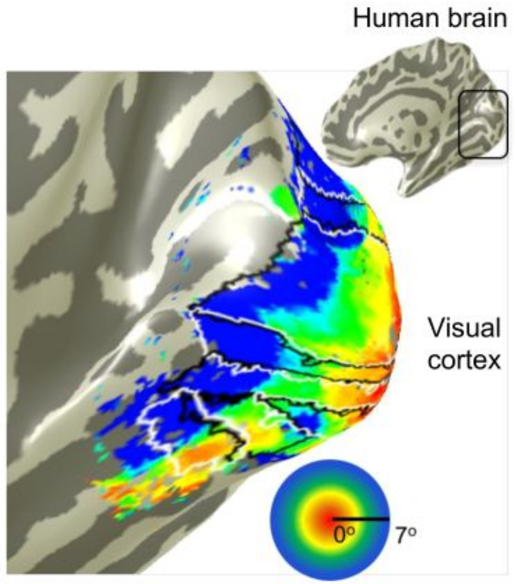
Cat image by CNX OpenStax is licensed under CC BY 4.0; changes made

METU

# Biological Background

The visual cortex is the primary cortical region of the brain that receives, integrates, and processes visual information relayed from the retinas.

It is in the occipital lobe of the primary cerebral cortex, which is in the most posterior region of the brain.



Retinotopy images courtesy of Jesse Gomez in the Stanford Vision & Perception Neuroscience Lab.

# CNNs vs. Visual Cortex

- Convolutional neural networks (CNNs) were inspired by early findings in the study of biological vision.
- The architecture of a CNN has (by design) direct parallels to the architecture of the visual system.

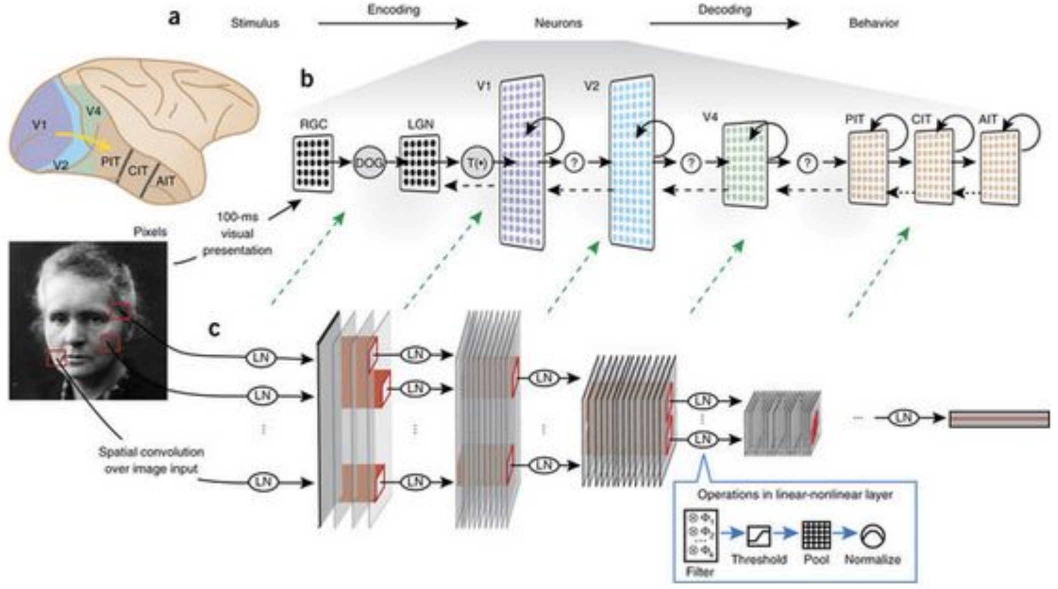
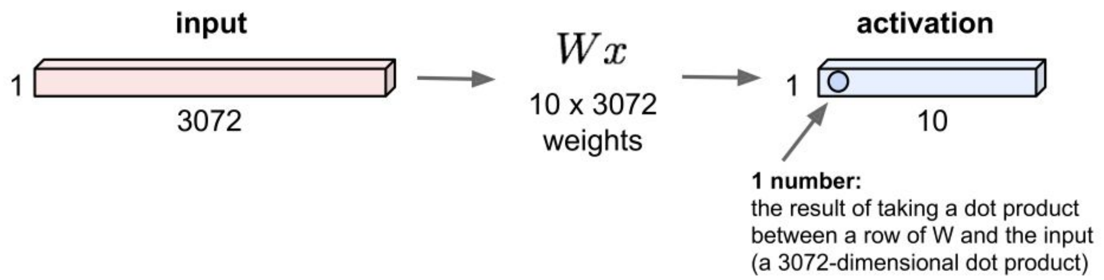


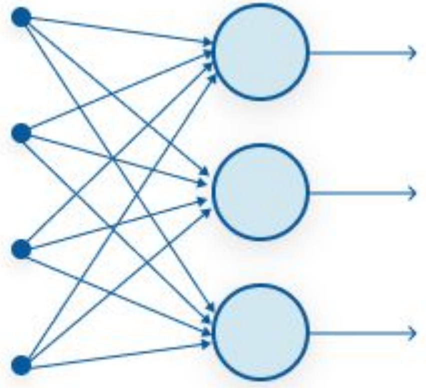
Figure from here.

METU

## Convolutional Neural Networks

- Previously we saw the ANN, in which every node was connected to every node at the next layer.
    - ANN, or Feed-forward Neural Network, or Multilayer perceptron, same.
- It is also called fully-connected layer:


input
1
3072
$Wx$
10 x 3072
weights
activation
1
10
**1 number:**
the result of taking a dot product between a row of W and the input (a 3072-dimensional dot product)

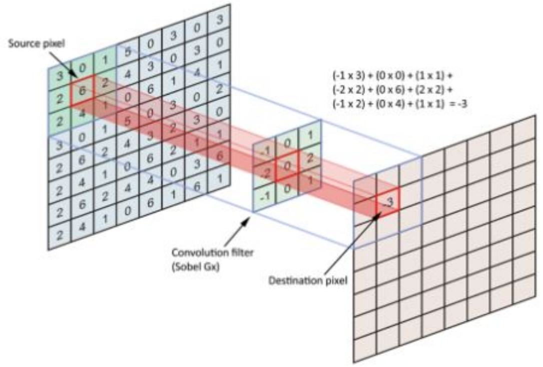


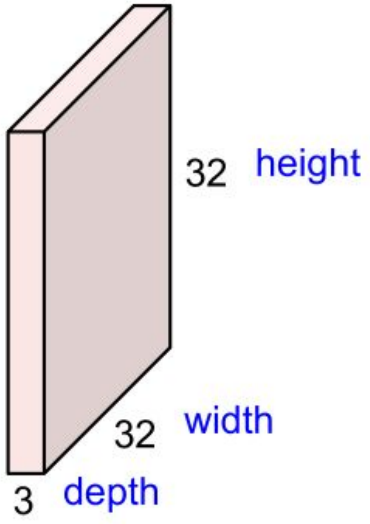
Feed-Forward Neural Network

METU

# Convolutional Neural Networks

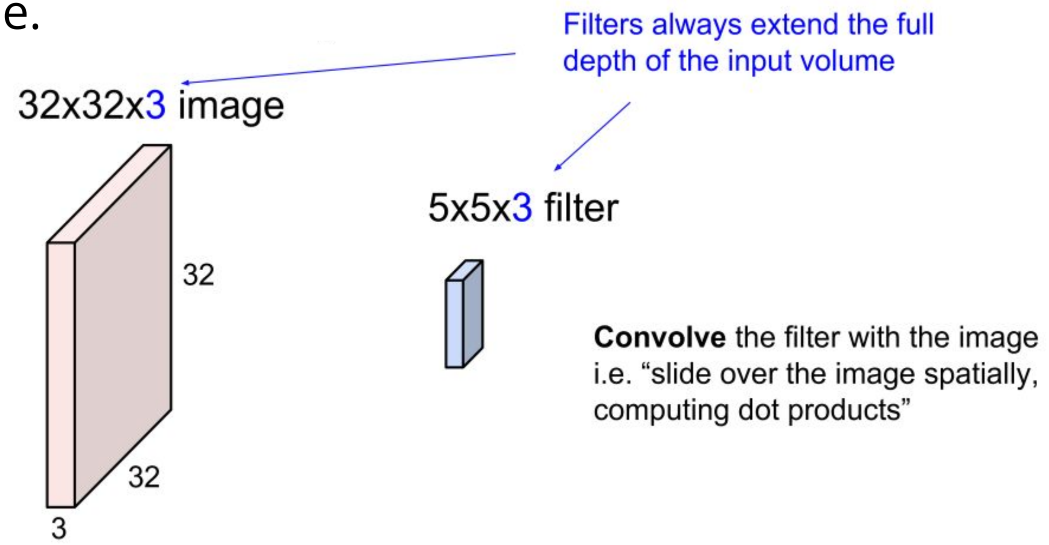
- In CNNs, every node is NOT connected.
- The input to a CNN has two “types” of dimensions “spatial” and “depth”.
- Spatial dimensions are the x,y,z coordinates of the signal
    - Image: 2D
    - Volume: 3D
- Depth dimension is the channel size
    - RGB : 3 channels


Source pixel
(-1 x 3) + (0 x 0) + (1 x 1) +
(-2 x 2) + (0 x 6) + (2 x 2) +
(-1 x 2) + (0 x 4) + (1 x 1) = -3
Convolution filter
(Sobel Gx)
Destination pixel



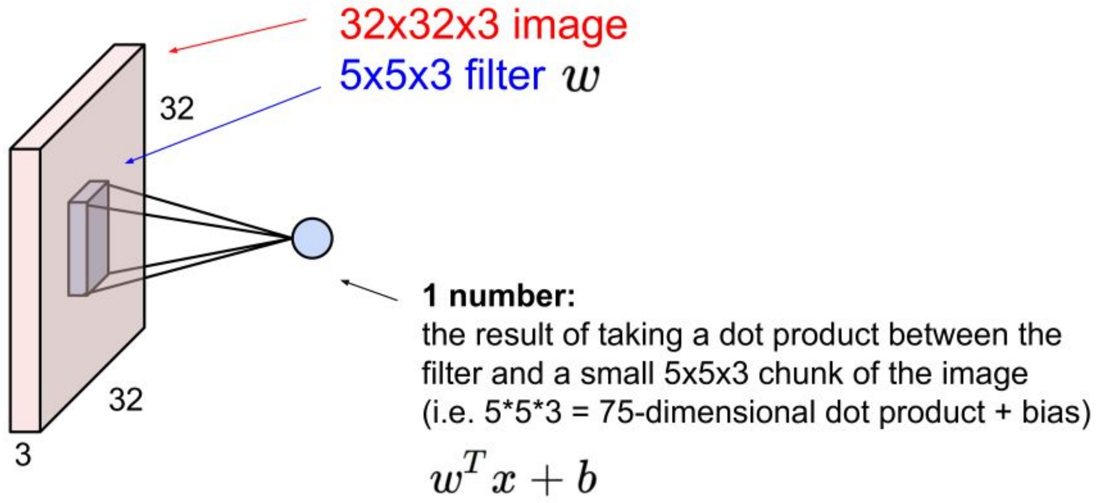
# Convolutional Neural Networks

- The filter has a smaller spatial dimension (usually)
    - If the spatial dims are the same, it becomes a fully-connected layer
- The depths must be the same.


Filters always extend the full depth of the input volume
32x32x3 image
32
32
3
5x5x3 filter
**Convolve** the filter with the image i.e. “slide over the image spatially, computing dot products”

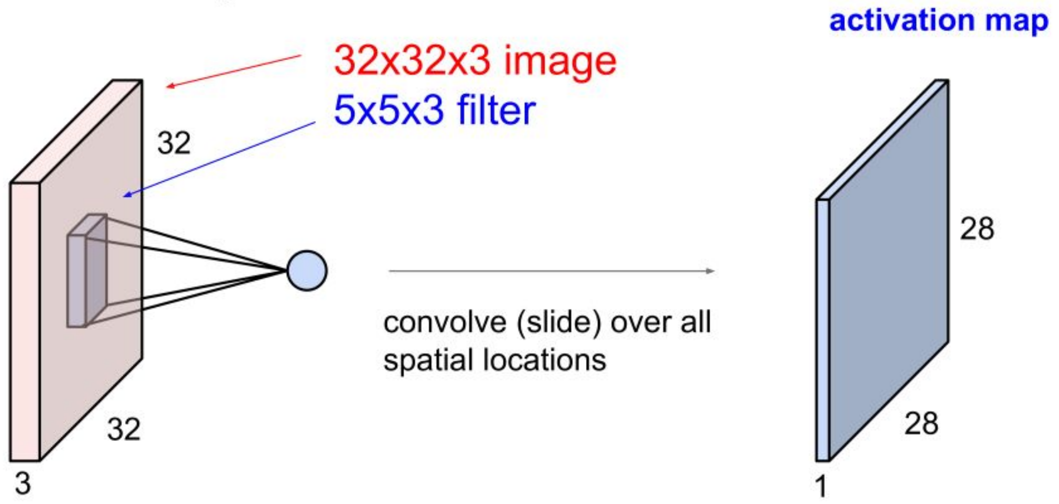
## Convolutional Neural Networks

- As discussed earlier, convolution operation is implemented as a scanning multiplication (with a bias).


32x32x3 image
5x5x3 filter $w$
32
32
3
**1 number:**
the result of taking a dot product between the filter and a small 5x5x3 chunk of the image
(i.e. 5*5*3 = 75-dimensional dot product + bias)
$w^T x + b$

# Convolutional Neural Networks

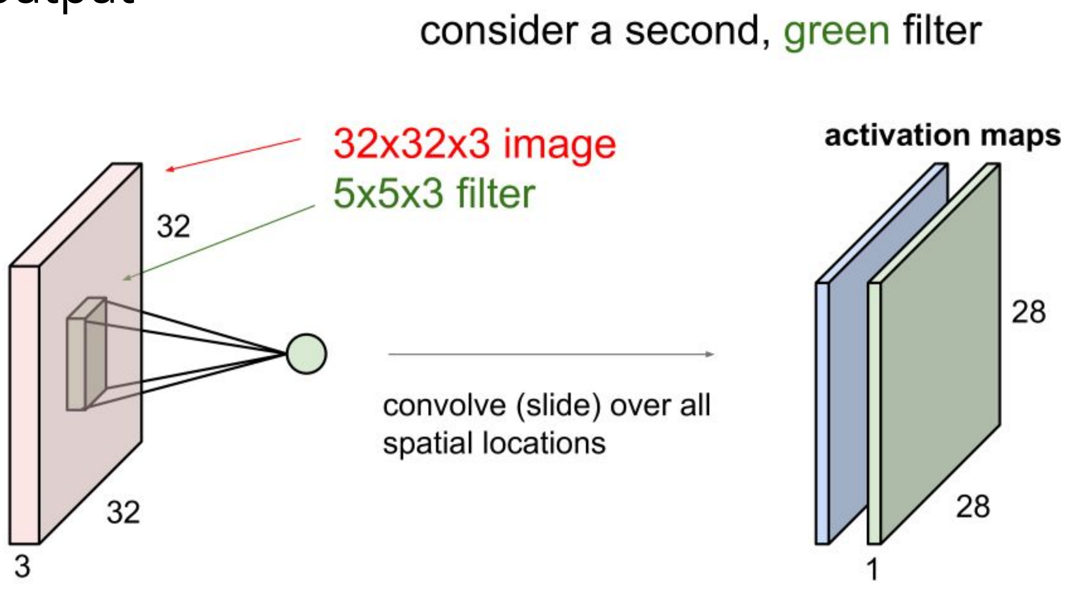
- When you “convolve” the filter all over the spatial dimensions, each sum creates an “activation”.



METU

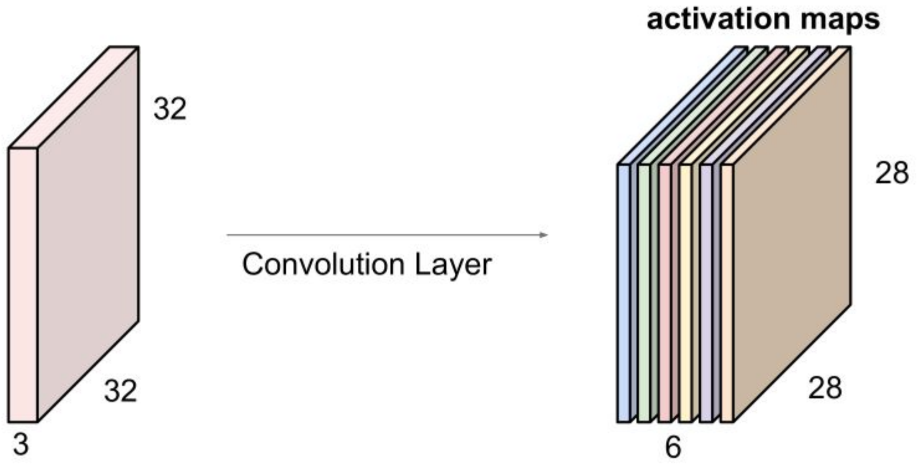
## Convolutional Neural Networks

- If there are N filters, the output depth becomes N


consider a second, green filter
32x32x3 image
5x5x3 filter
32
32
3
convolve (slide) over all spatial locations
activation maps
28
28
1

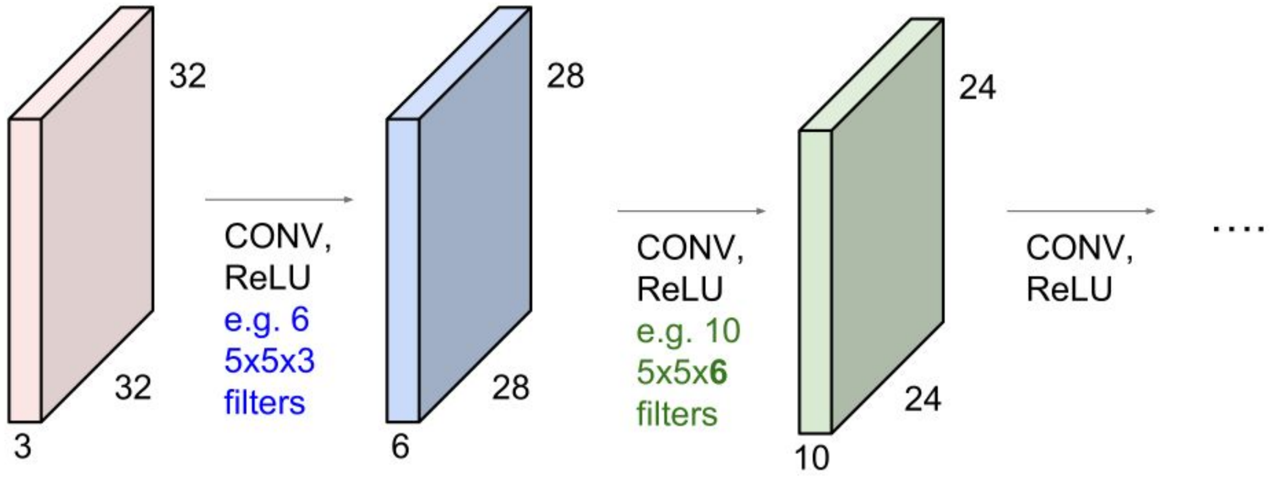
# Convolutional Neural Networks

- If there are N filters, the output depth becomes N
  - Input 32x32x**3**
  - Filter 5x5x**3**x**6**
  - Output 28x28x**6**



## Convolutional Layer

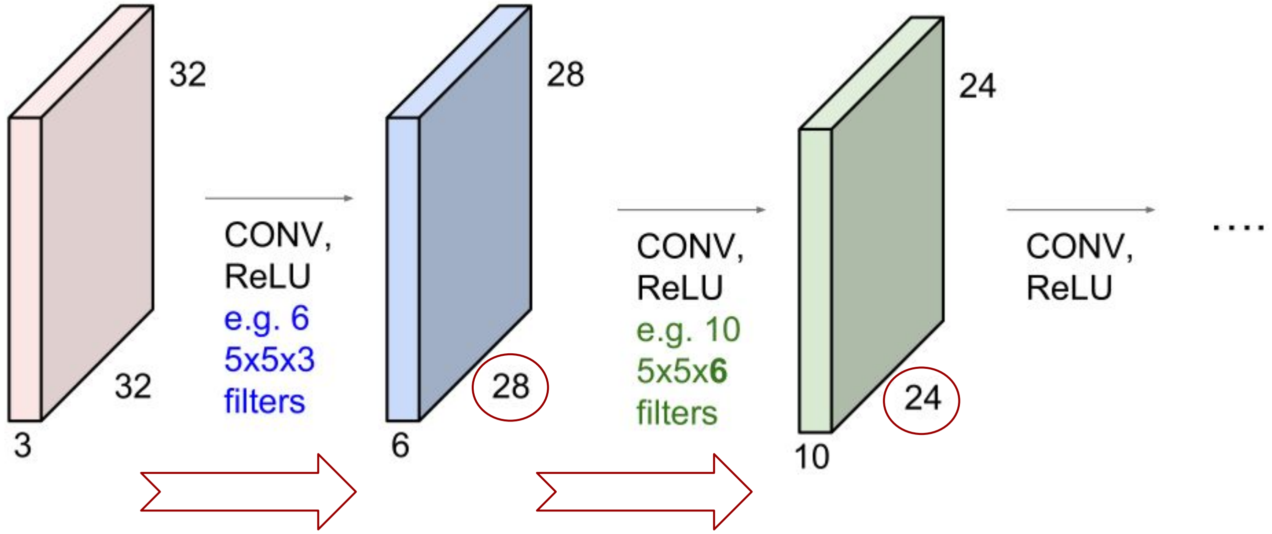
- And each filter creates a “layer”.



METU

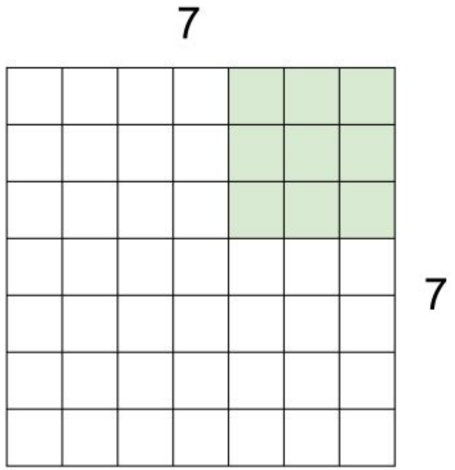
# Convolutional Layer

- And each filter creates a “layer”.



## Convolutional Layer Output

- In order to fit the filter, at the edge points, activations cannot be calculated.

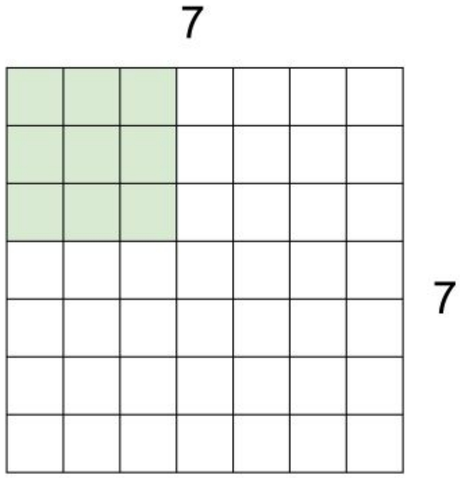


7x7 input (spatially)
assume 3x3 filter

**=> 5x5 output**

# Convolutional Layer Output with Stride

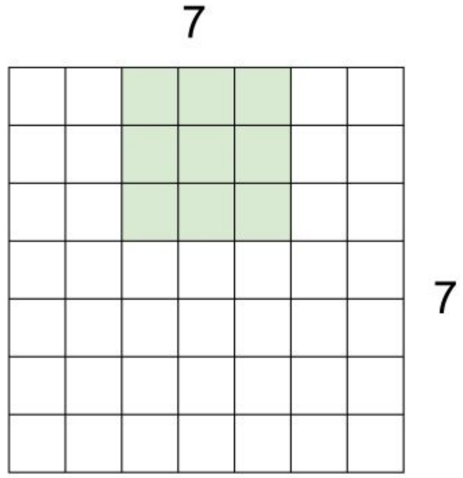
- Stride is the “scanning jump size”.



7x7 input (spatially)
assume 3x3 filter
applied **with stride 2**

# Convolutional Layer Output with Stride

- Stride is the "scanning jump size".

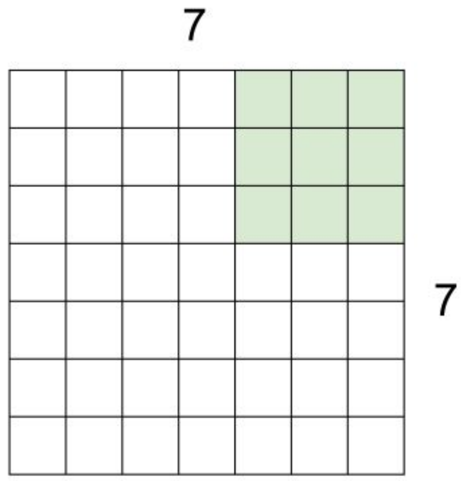


7x7 input (spatially)
assume 3x3 filter
applied **with stride 2**

METU

## Convolutional Layer Output with Stride

- Stride is the “scanning jump size”.

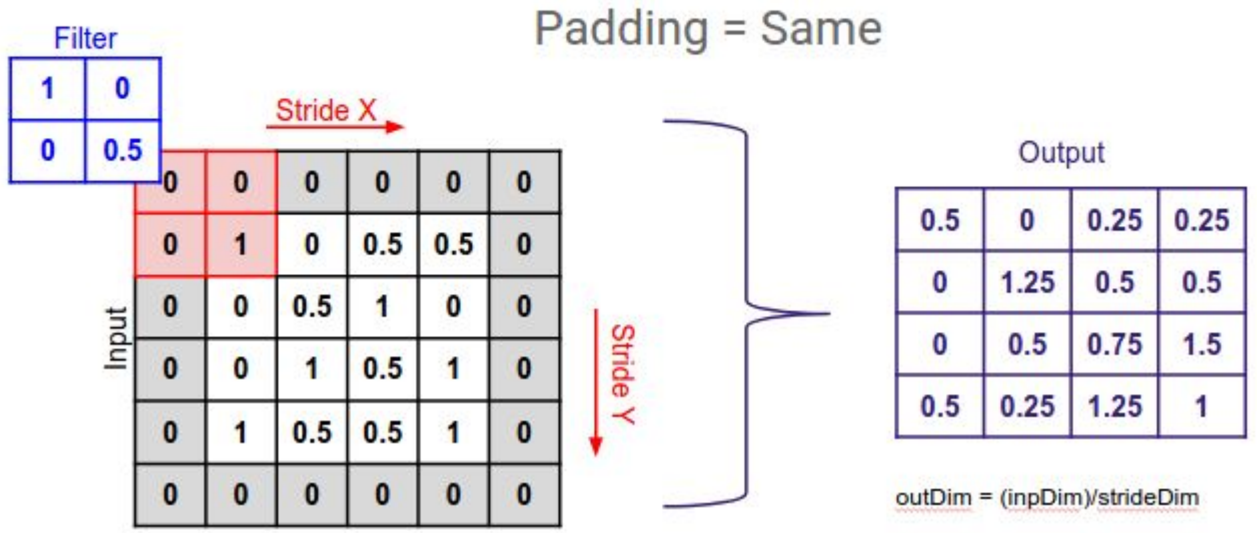


7x7 input (spatially)
assume 3x3 filter
applied **with stride 2**
**=> 3x3 output!**

METU

# Convolutional Layer Output with Padding

- Padding is the “edge enlargement”.



| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0.5 | 0.5 | 0 |
| 0 | 0 | 0.5 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0.5 | 1 | 0 |
| 0 | 1 | 0.5 | 0.5 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |

| | | | |
|---|---|---|---|
| 0.5 | 0 | 0.25 | 0.25 |
| 0 | 1.25 | 0.5 | 0.5 |
| 0 | 0.5 | 0.75 | 1.5 |
| 0.5 | 0.25 | 1.25 | 1 |

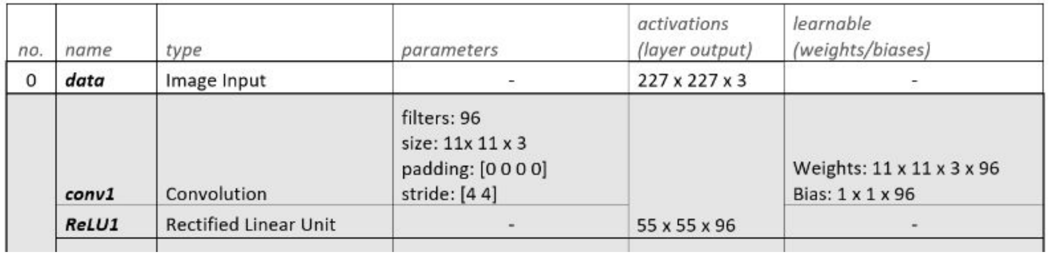
# Convolutional Layer Output

- The output size of a **2D** convolutional layer with input $\boldsymbol{M \times N}$ and with filter size $\boldsymbol{k \times l \times F}$, with padding of P (on both sides) and a stride of S, will be $\boldsymbol{W \times H \times F}$, and can be easily defined with a simple formula:

$$W = \frac{M - k + 2P}{S} + 1, \; H = \frac{N - l + 2P}{S} + 1,$$

METU

# Practice on AlexNet



| no. | name | type | parameters | activations (layer output) | learnable (weights/biases) |
|---|---|---|---|---|---|
| 0 | ***data*** | Image Input | - | 227 x 227 x 3 | - |
| | ***conv1*** | Convolution | filters: 96<br>size: 11x 11 x 3<br>padding: [0 0 0 0]<br>stride: [4 4] | | Weights: 11 x 11 x 3 x 96<br>Bias: 1 x 1 x 96 |
| | ***ReLU1*** | Rectified Linear Unit | - | 55 x 55 x 96 | - |

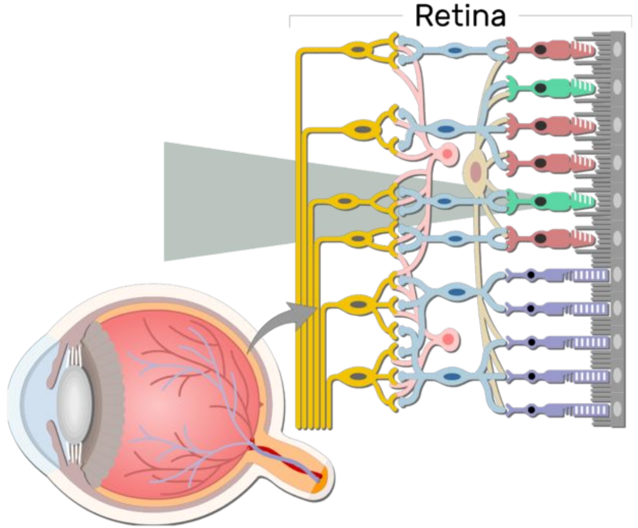
- The famous AlexNet gets input of 227x227x3 RGB images.
- This input is fed to the first convolutional layer (i.e. conv1) with 96 filters, each filter having a size of 11x11x3.
- This is a 2D conv filter, so that it moves along the images, and convolves with a 11x11x3 RGB patch at each operation.
- Thus the learnable parameters in this layer is 11x11x3x96 (34,848) number of weights, plus 96 biases (a single bias for each filter. Thus this layer includes 34,848+96=34,944 learnable values.
- Because the stride is 4 and there is no padding in “conv1”, we span each filter 55 x 55 ((227-11-0)/4 +1 = 55) number of times, and thus the output of this layer is 55x55x96, which is the input to the next layer.

$$W = \frac{M-k+2P}{S} + 1, W = \frac{N-l+2P}{S} + 1,$$

METU

# Receptive Field

- In biology, the receptive field, or sensory space, is a delimited medium where some physiological stimuli can evoke a sensory neuronal response in specific organisms.



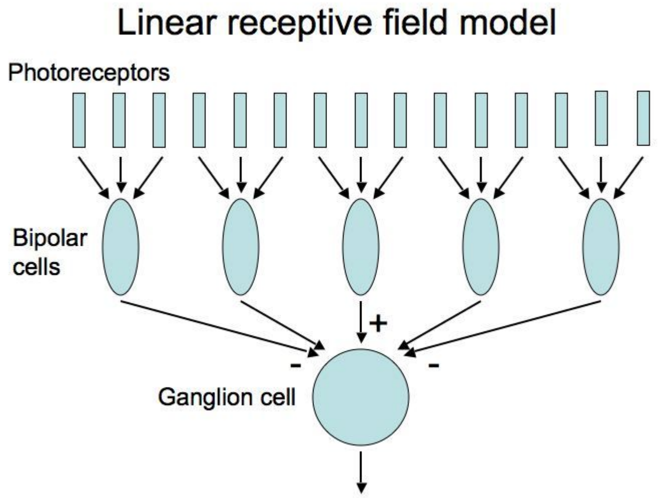
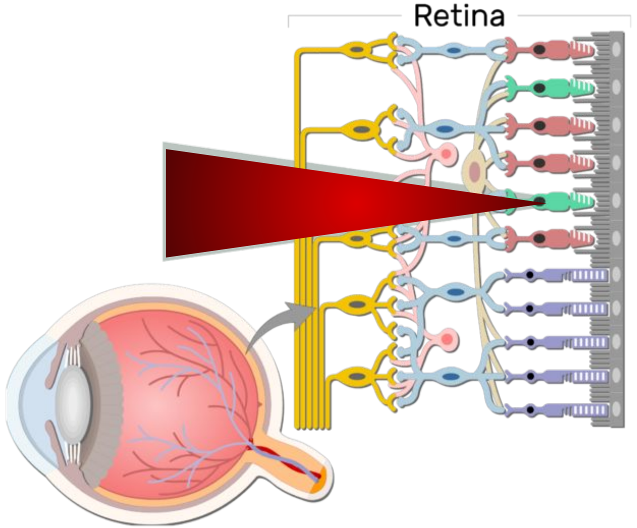


Figure from here.

METU

# Receptive Field

- In biology, the receptive field, or sensory space, is a delimited medium where some physiological stimuli can evoke a sensory neuronal response in specific organisms.



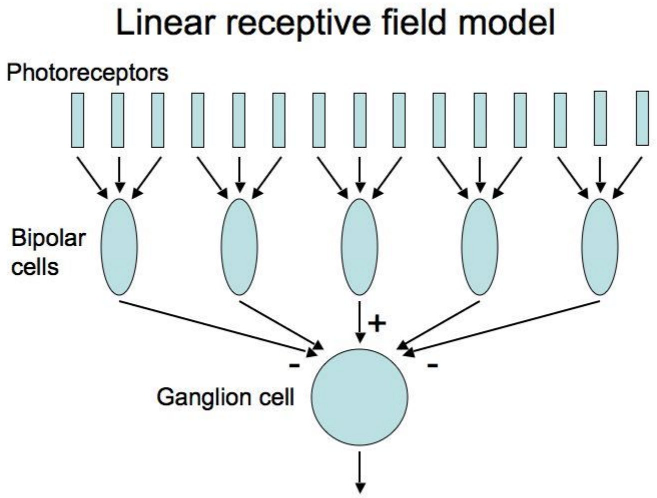
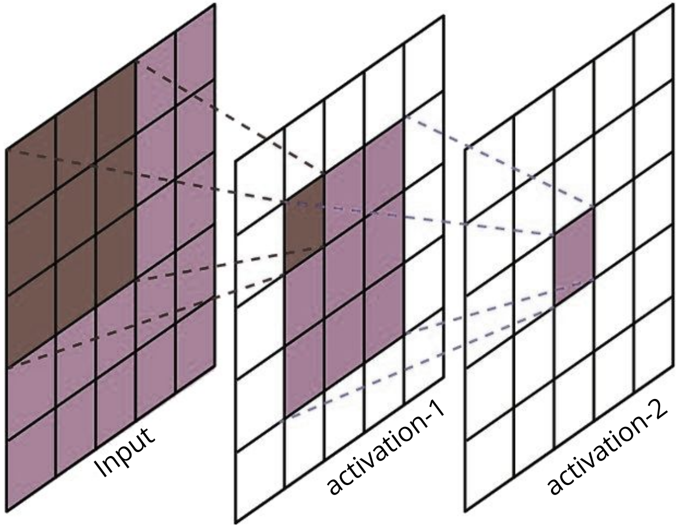


Figure from here.

METU

## Receptive Field of a Conv. Net



- Imagine we have a 2-layered conv. net with:
  - filters 3x3 filters at each layer.
  - Input 5x5
  - No padding, no stride.
- Since there is no padding, activation after layer-2 is 1x1.
- The receptive field is defined as the region in the input space that a particular CNN's activation is looking at (i.e. be affected by) .
- A receptive field of a feature can be described by its center location and its size.

# Pooling Layers

- Convolutional layers in a convolutional neural network systematically apply learned filters to input images in order to create feature maps that summarize the presence of those features in the input.
-

## Pooling Layers

- Convolutional layers in a convolutional neural network systematically apply learned filters to input images in order to create feature maps that summarize the presence of those features in the input.
- Convolutional layers prove very effective, and stacking convolutional layers in deep models allows layers close to the input to learn low-level features (e.g. lines) and layers deeper in the model to learn high-order or more abstract features, like shapes or specific objects.
-

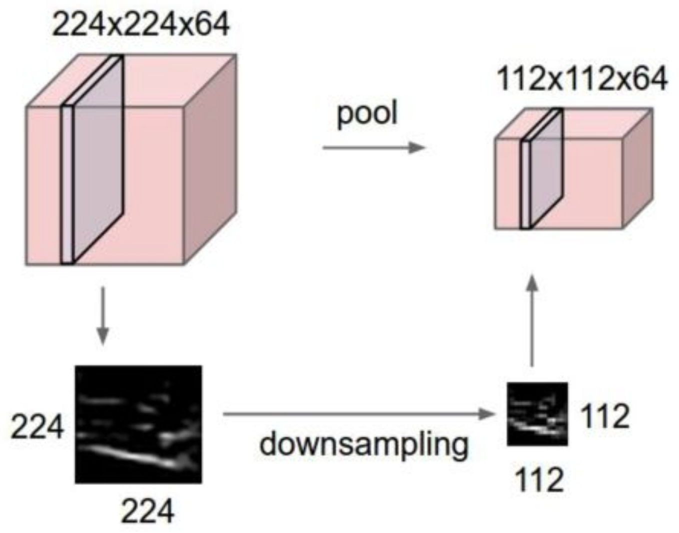
## Pooling Layers

- Convolutional layers in a convolutional neural network systematically apply learned filters to input images in order to create feature maps that summarize the presence of those features in the input.
- Convolutional layers prove very effective, and stacking convolutional layers in deep models allows layers close to the input to learn low-level features (e.g. lines) and layers deeper in the model to learn high-order or more abstract features, like shapes or specific objects.
- A limitation of the feature map output of convolutional layers is that they record the precise position of features in the input. This means that small movements in the position of the feature in the input image will result in a different feature map. This can happen with re-cropping, rotation, shifting, and other minor changes to the input image.

METU

# Pooling Layers

- A common approach to addressing this problem from signal processing is called down sampling.
- This is where a lower resolution version of an input signal is created that still contains the large or important structural elements, without the fine detail that may not be as useful to the task.



## Pooling Layers

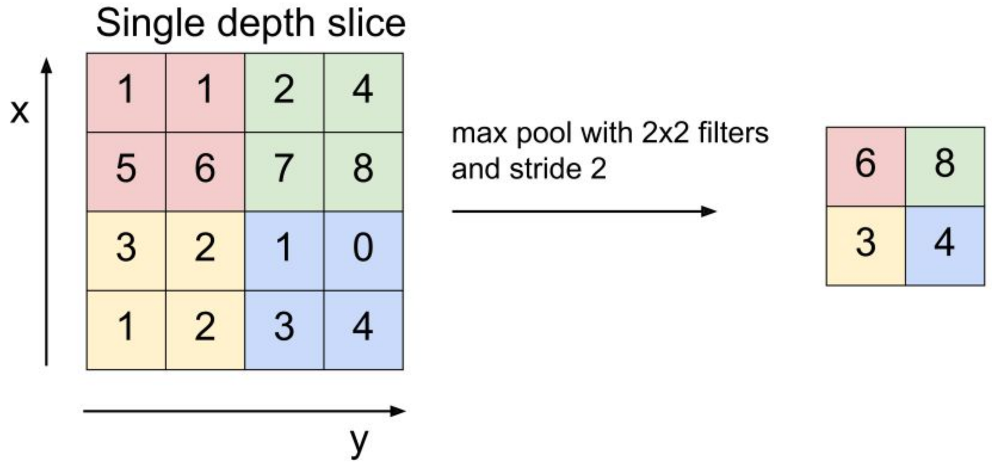
- There are different ways for pooling (down-sampling):
  - Average Pooling: Calculate the average value for each patch on the feature map.
  - Maximum Pooling (or Max Pooling): Calculate the maximum value for each patch of the feature map.
  - Minimum Pooling (not used much
  - etc.

## Pooling Layers

- There are different ways for pooling (down-sampling):
    - Average Pooling: Calculate the average value for each patch on the feature map.
    - Maximum Pooling (or Max Pooling): Calculate the maximum value for each patch of the feature map.
    - Minimum Pooling (not used much
    - etc.

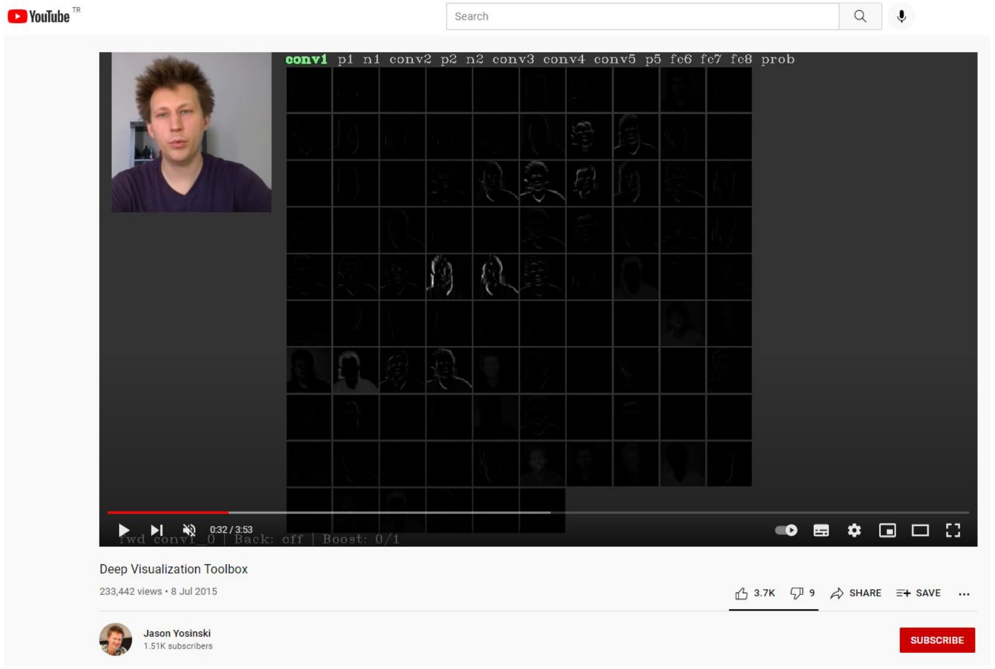
## Max Pooling

- Calculate the maximum value for each patch of the feature map.



## Live Code!

- Let's go over the entire AlexNet and let's visualise some activations (features).
- For this implementation I will be using MATLAB. But there are ways to this on Python as well.
- Then watch this: https://www.youtube.com/watch?v=AgkfIQ4IGaM


YouTube TR
Search
conv1 p1 n1 conv2 p2 n2 conv3 conv4 conv5 p5 fc6 fc7 fc8 prob
fwd conv1_0 | Back: off | Boost: 0/1
0:32 / 3:53
Deep Visualization Toolbox
233,442 views • 8 Jul 2015
3.7K
9
SHARE
SAVE
Jason Yosinski
1.51K subscribers
SUBSCRIBE

## What to do until next week?

- You already (!) have a working DL environment.
- Think of a project idea. Preferably related to your thesis.
  - ➢ You don’t know how to do it yet, even if it can be done or not. Do some research. Did people do it before? What could they achieve? Maybe you will implement a modification on the problem? Start thinking about it.
  - ➢ Start preparing an abstract on your project idea and next time, ask for feedback.

## What will we do next week?

- Starting with next week...
  - Continue Conv. Nets, Convolutional Encoders, Semantic Segmentation...
- Following weeks
  - Sequence Models, Introduction to NLP ...

## Additional Reading & References

- https://arxiv.org/ftp/arxiv/papers/2001/2001.07092.pdf
- https://embryo.asu.edu/pages/david-h-hubel-and-torsten-n-wiesels-research-optical-development-kittens
- https://becominghuman.ai/from-human-vision-to-computer-vision-convolutional-neural-network-part3-4-24b55ffa7045
- https://www.cns.nyu.edu/~david/courses/perception/lecturenotes/ganglion/ganglion.html
- https://christophm.github.io/interpretable-ml-book/cnn-features.html
- https://www.mathworks.com/help/deeplearning/ug/visualize-activations-of-a-convolutional-neural-network.html
- https://www.mathworks.com/help/deeplearning/ug/visualize-activations-of-a-convolutional-neural-network.html
- https://github.com/utkuozbulak/pytorch-cnn-visualizations
- https://www.youtube.com/watch?v=AgkfIQ4IGaM