Domain-specific networks
Convolutional Neural Networks

Dr. Eran Raviv

July 2022

Agenda for today

- Recap from last time: what have we learned?

Agenda for today

- Recap from last time: what have we learned?
- Domain specific models.

Convolutional Neural Networks

- What is an image?
- Why are they so relevant in Computer Vision?
- What does convolution actually mean?
- What are convolutional neural networks?

Convolutional neural networks are domain-aware. They are the way to go when it comes to computer vision.

True or false

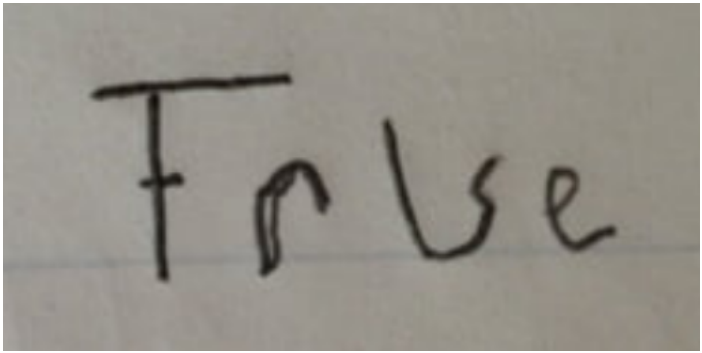
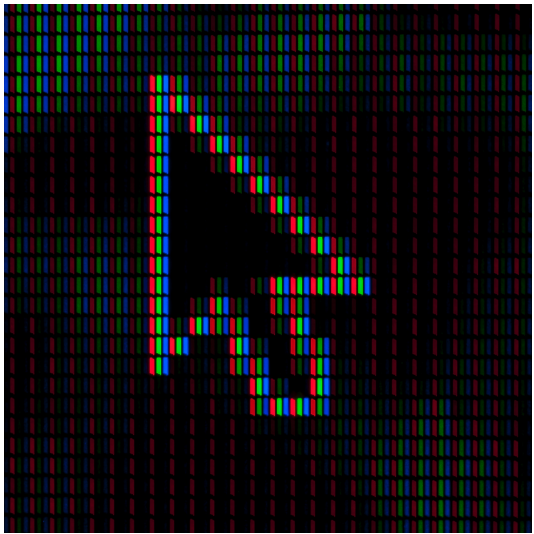
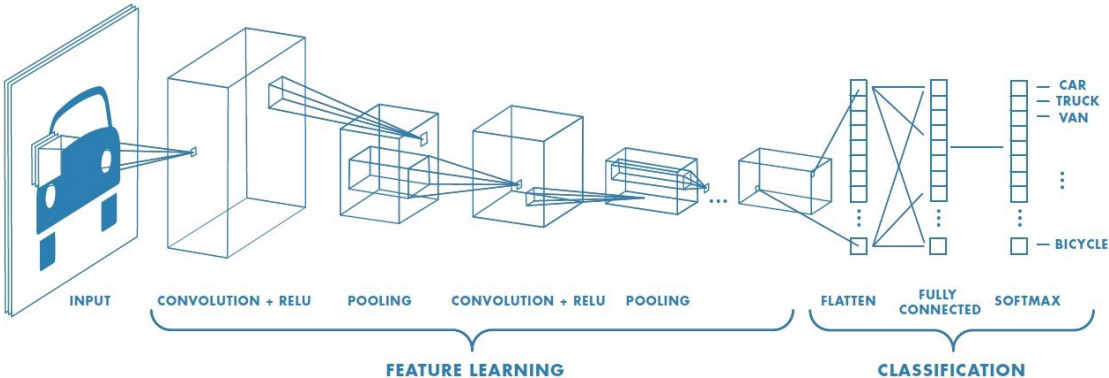


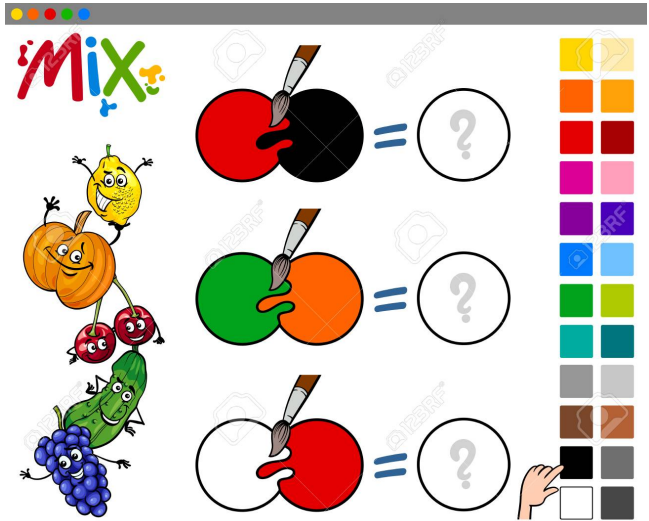
Image representation



Convolutional neural networks - first look



What is convolution?



What is convolution?

Convolution is a mathematical operation on two functions to produce a third function. The term convolution refers to both the result function and to the process of computing it.

The convolution operator

- The convolution operation:

$$s(t) = (y * w)(t)$$

The convolution operator

- The convolution operation:

$$s(t) = (y * w)(t)$$

- For example:

$$s(x) = \int y(t)w(x-t)dt$$

but for simplicity, you can think of about it as:

$$\text{EMA}_t = \alpha y_t + \alpha(1-\alpha)y_{t-1} + \alpha(1-\alpha)^2 y_{t-2} + \ldots + \alpha(1-\alpha)^t y_0 =$$

$$\alpha \sum_{i=0}^{t} (1-\alpha)^{t-i} y_{t-i}$$

The convolution operator in two dimensions

For functions with two dimensions- which is what is relevant for us:

$$S(i,j) := (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m,j-n)$$

Or equivalently:

$$S(i,j) := (K * I)(i,j) = \sum_m \sum_n I(i-m,j-n)K(m,n)$$

The convolution operator

- *Scalar* $\times$ *Matrix* is a private, trivial case of the convolution operator.

The convolution operator

- *Scalar* $\times$ *Matrix* is a private, trivial case of the convolution operator.
- We will need it this triviality when working with higher dimensions.

The convolution operator

- *Scalar* $\times$ *Matrix* is a private, trivial case of the convolution operator.
- We will need it this triviality when working with higher dimensions.
- In 2012, Krizhevsky et al. rolled out the red carpet for the Deep Convolutional Neural Network.

The convolution operator

- *Scalar* $\times$ *Matrix* is a private, trivial case of the convolution operator.
- We will need it this triviality when working with higher dimensions.
- In 2012, Krizhevsky et al. rolled out the red carpet for the Deep Convolutional Neural Network.
- Well suited for image data (why?)

Image processing operator

Image processing operator

Image processing operator

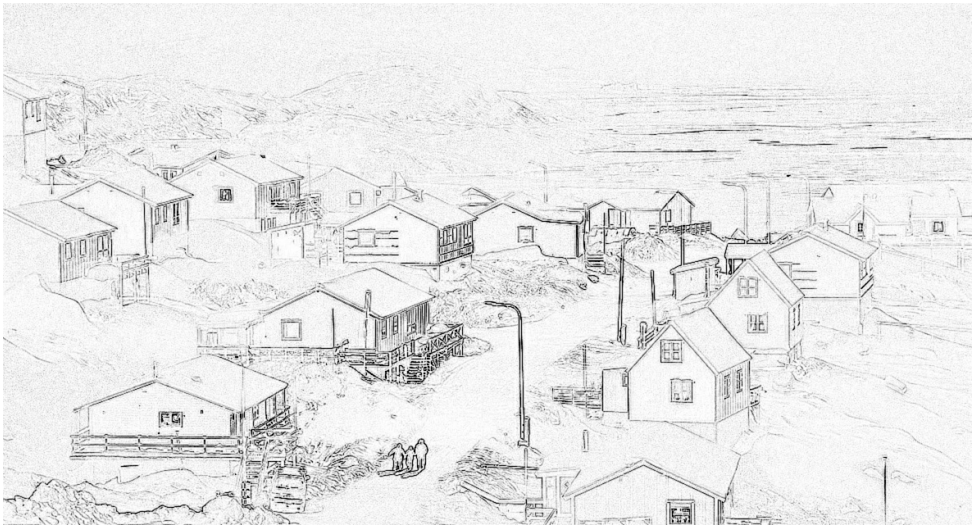


Image processing operator

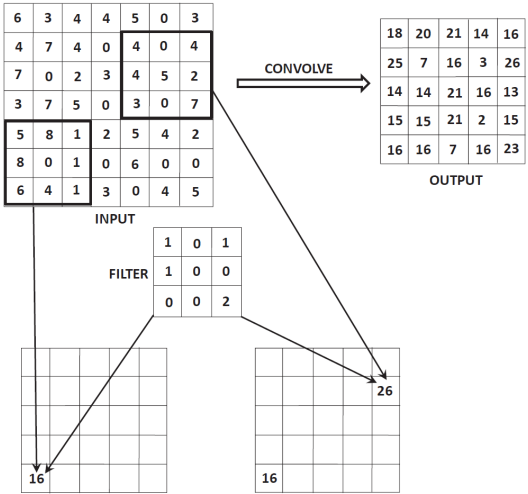
What is the convolution here?

Add G to the RGB's

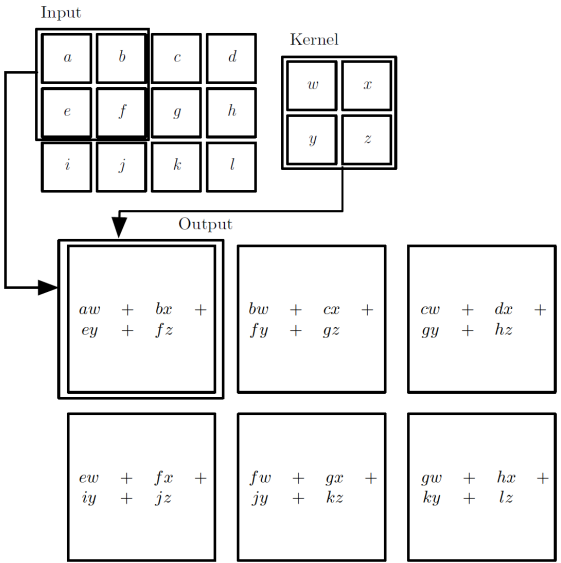
Only the saturation channel

Is this a different village?

In two dimensions



In two dimensions



- The Kernel is what we need to teach the machine.
- Sometimes this is referred to also as a Cross-Correlation Operator, and the kernel is referred to as a correlation kernel.
- We will use the more general *filter* term (stacked multiple kernels).

In higher dimension

Recall that a colored image can be represented using RGB. We can have different kernels/filters for each channel separately.

- Practically, the dimension of the output does not change.

In higher dimension

Recall that a colored image can be represented using RGB. We can have different kernels/filters for each channel separately.

- Practically, the dimension of the output does not change.
- The reason is that each entry is now the sum over all channels.

In higher dimension

Recall that a colored image can be represented using RGB. We can have different kernels/filters for each channel separately.

- Practically, the dimension of the output does not change.
- The reason is that each entry is now the sum over all channels.
- But we have multiple filter for better flexibility and expressiveness.

Padding and Striding, and Filtering

Padding

- Convolution operation reduces the size of the next layer compare to current layer.

Padding

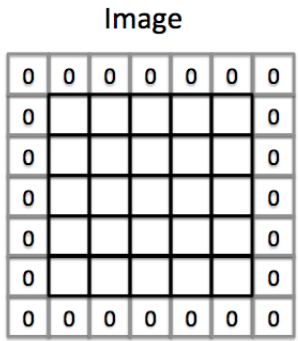
- Convolution operation reduces the size of the next layer compare to current layer.
- Edges are under represented.

Padding

- Convolution operation reduces the size of the next layer compare to current layer.
- Edges are under represented.

Padding

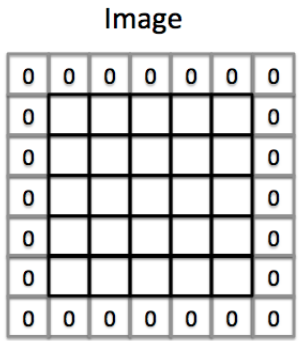
- Convolution operation reduces the size of the next layer compare to current layer.
- Edges are under represented.



- In order to solve those two disadvantages we pad.

Padding

- Convolution operation reduces the size of the next layer compare to current layer.
- Edges are under represented.



- In order to solve those two disadvantages we pad.
- Padding is nothing more than adding "blank" pixels around all borders of the feature map.

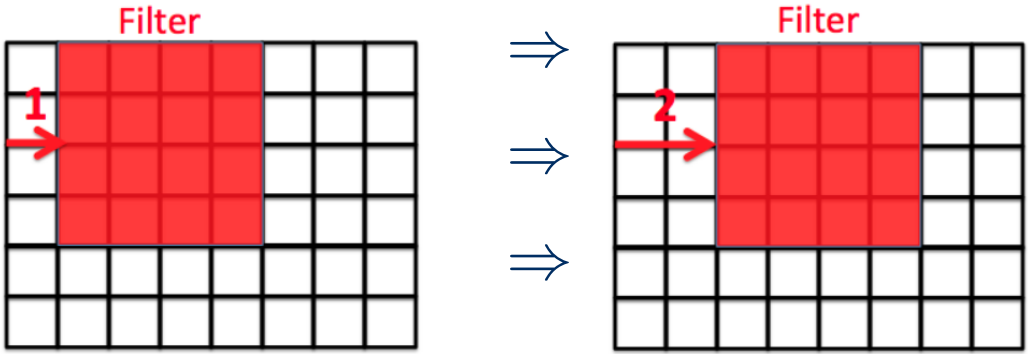
Padding

- *Valid:* No padding

Padding

- *Valid:* No padding
- *Same:* $\text{pad\_size} = \frac{\text{filter\_size} - 1}{2}$

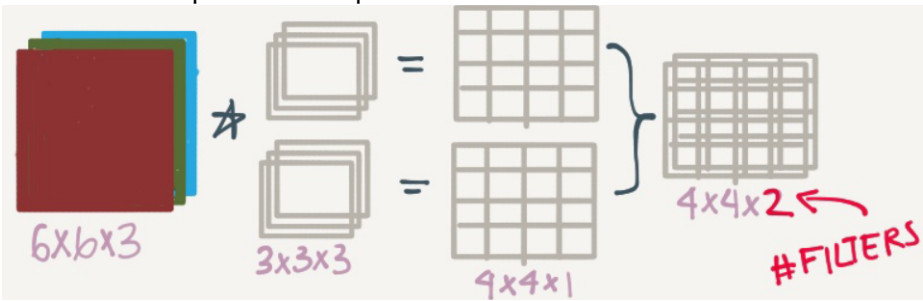
Striding



Simply: stride is the number of steps we traverse across the image (horizontally as well as vertically). Decrease computational cost.

Convolution using multiple filters

- Now the output size depends on the number of filters:



- We sum them up element-wise to form a **feature map** before we "send it right".
- Each filter has $3 \times 3 \times 3$ parameters to be estimated. In this figure a total of 54 parameters (2 filters).

Advantages of convolutions

- Parameter sharing - computational efficiency (compare fully connected to convolutions).

Advantages of convolutions

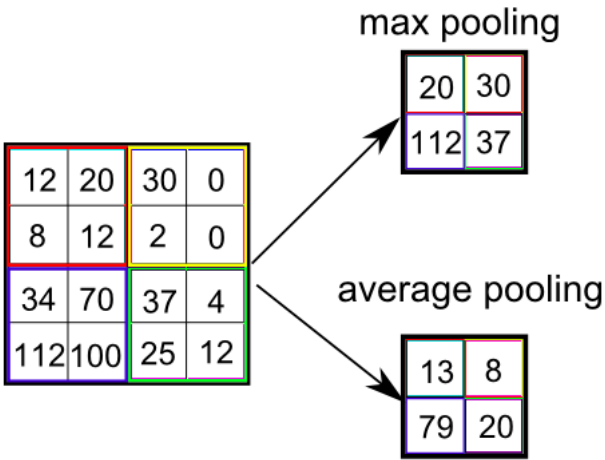
- Parameter sharing - computational efficiency (compare fully connected to convolutions).
- Parameter sharing - an eye is an eye is an eye. 

Advantages of convolutions

- Parameter sharing - computational efficiency (compare fully connected to convolutions).
- Parameter sharing - an eye is an eye is an eye. 
- Sparsity of connections.

Pooling layer

Apply a pooling function.



Pooling layer

- No parameters, only hyperparameters.

Pooling layer

- No parameters, only hyperparameters.
- As summary of the inputs over a small neighborhood.

Pooling layer

- No parameters, only hyperparameters.
- As summary of the inputs over a small neighborhood.
- Just a different representation of of the data. E.g. caring about the average tone, or about a screaming tone.

Pooling layer

- No parameters, only hyperparameters.
- As summary of the inputs over a small neighborhood.
- Just a different representation of of the data. E.g. caring about the average tone, or about a screaming tone.
- Not much theory behind it.

Pooling layer

- No parameters, only hyperparameters.
- As summary of the inputs over a small neighborhood.
- Just a different representation of of the data. E.g. caring about the average tone, or about a screaming tone.
- Not much theory behind it.
- Exists in mixed frequency time series as well: last quote, or average quote?

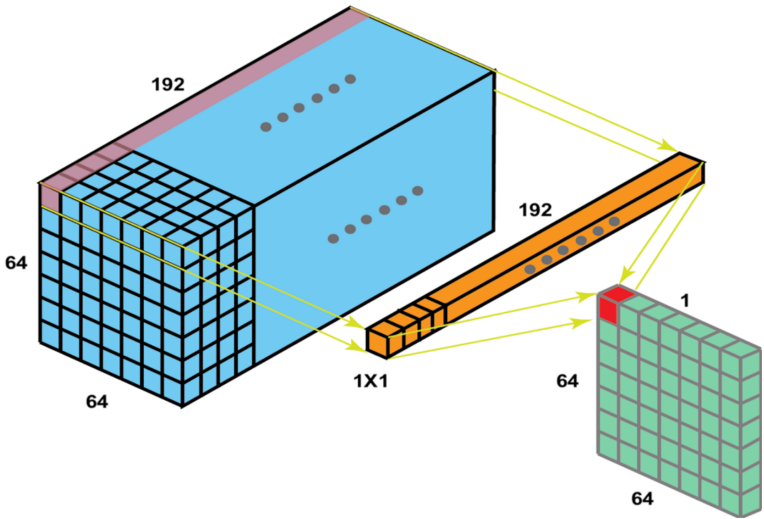
Pooling layer

- No parameters, only hyperparameters.
- As summary of the inputs over a small neighborhood.
- Just a different representation of of the data. E.g. caring about the average tone, or about a screaming tone.
- Not much theory behind it.
- Exists in mixed frequency time series as well: last quote, or average quote?
- Max-pooling is more common than average pooling.

Use case

Few "tricks of the trade"

Network in Network



Inception Networks

Szegedy et al. 2015

> *(Inception Layer) is a combination of all those layers (namely, 1×1 Convolutional layer, 3×3 Convolutional layer, 5×5 Convolutional layer) with their output filter banks concatenated into a single output vector forming the input of the next stage.*

- Let the network learn from different granularity levels.

Inception Networks

Szegedy et al. 2015

> *(Inception Layer) is a combination of all those layers (namely, 1×1 Convolutional layer, 3×3 Convolutional layer, 5×5 Convolutional layer) with their output filter banks concatenated into a single output vector forming the input of the next stage.*

- Let the network learn from different granularity levels.
- Where else did you see this idea?

Inception Networks

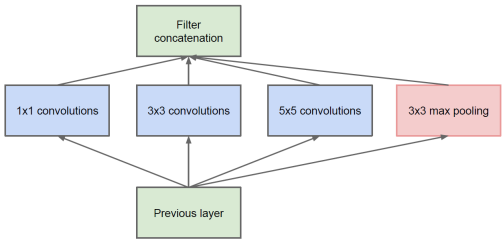
Szegedy et al. 2015

> *(Inception Layer) is a combination of all those layers (namely, 1×1 Convolutional layer, 3×3 Convolutional layer, 5×5 Convolutional layer) with their output filter banks concatenated into a single output vector forming the input of the next stage.*

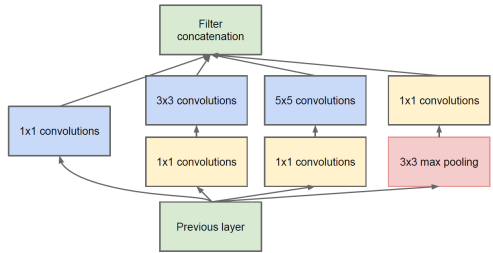
- Let the network learn from different granularity levels.
- Where else did you see this idea?
- Currently(!) one of the best performing CNNs, especially in computer vision.

GoogLeNet - Inception networks

naïve version

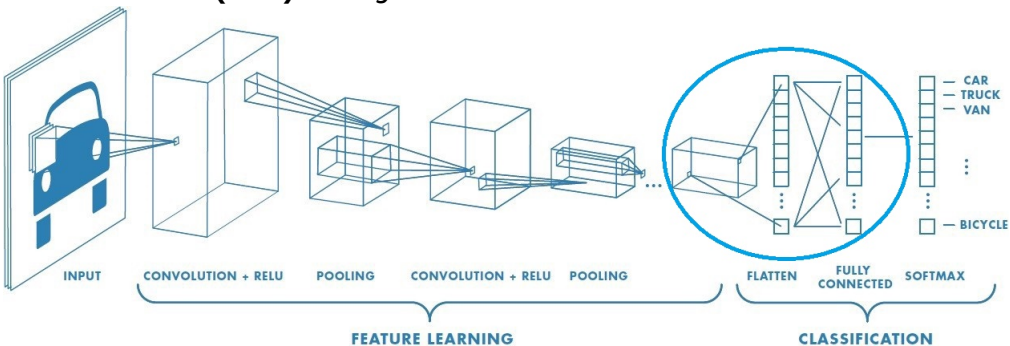


With dimension reduction



Global Average Pooling

(too) Busy chunk of the network:



Global Average Pooling

- Traditionally, after the "engineering step" is done the feature map is flattened and passed on to one (or more) fully connected layer, before heading to the softmax function.

Global Average Pooling

- Traditionally, after the "engineering step" is done the feature map is flattened and passed on to one (or more) fully connected layer, before heading to the softmax function.
- A fully connected layer has many parameters and hence adds **training complexity**, and, is prone to overfitting.

Global Average Pooling

- Traditionally, after the "engineering step" is done the feature map is flattened and passed on to one (or more) fully connected layer, before heading to the softmax function.
- A fully connected layer has many parameters and hence adds **training complexity**, and, is prone to overfitting.
- In Lin et al. 2013 the authors propose to global-average the last channel. That average score (each channel corresponds with a class) is fed directly to the softmax layer.

Global Average Pooling

- Traditionally, after the "engineering step" is done the feature map is flattened and passed on to one (or more) fully connected layer, before heading to the softmax function.
- A fully connected layer has many parameters and hence adds **training complexity**, and, is prone to overfitting.
- In Lin et al. 2013 the authors propose to global-average the last channel. That average score (each channel corresponds with a class) is fed directly to the softmax layer.
  - Direct mapping between engineered features and the score is more intuitive.

Global Average Pooling

- Traditionally, after the "engineering step" is done the feature map is flattened and passed on to one (or more) fully connected layer, before heading to the softmax function.
- A fully connected layer has many parameters and hence adds **training complexity**, and, is prone to overfitting.
- In Lin et al. 2013 the authors propose to global-average the last channel. That average score (each channel corresponds with a class) is fed directly to the softmax layer.
  - Direct mapping between engineered features and the score is more intuitive.
  - **Much** less parameters to train.

References

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25:1097–1105.

Lin, M., Chen, Q., and Yan, S. (2013). Network in network. arXiv preprint arXiv:1312.4400.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1–9.