Deep learning

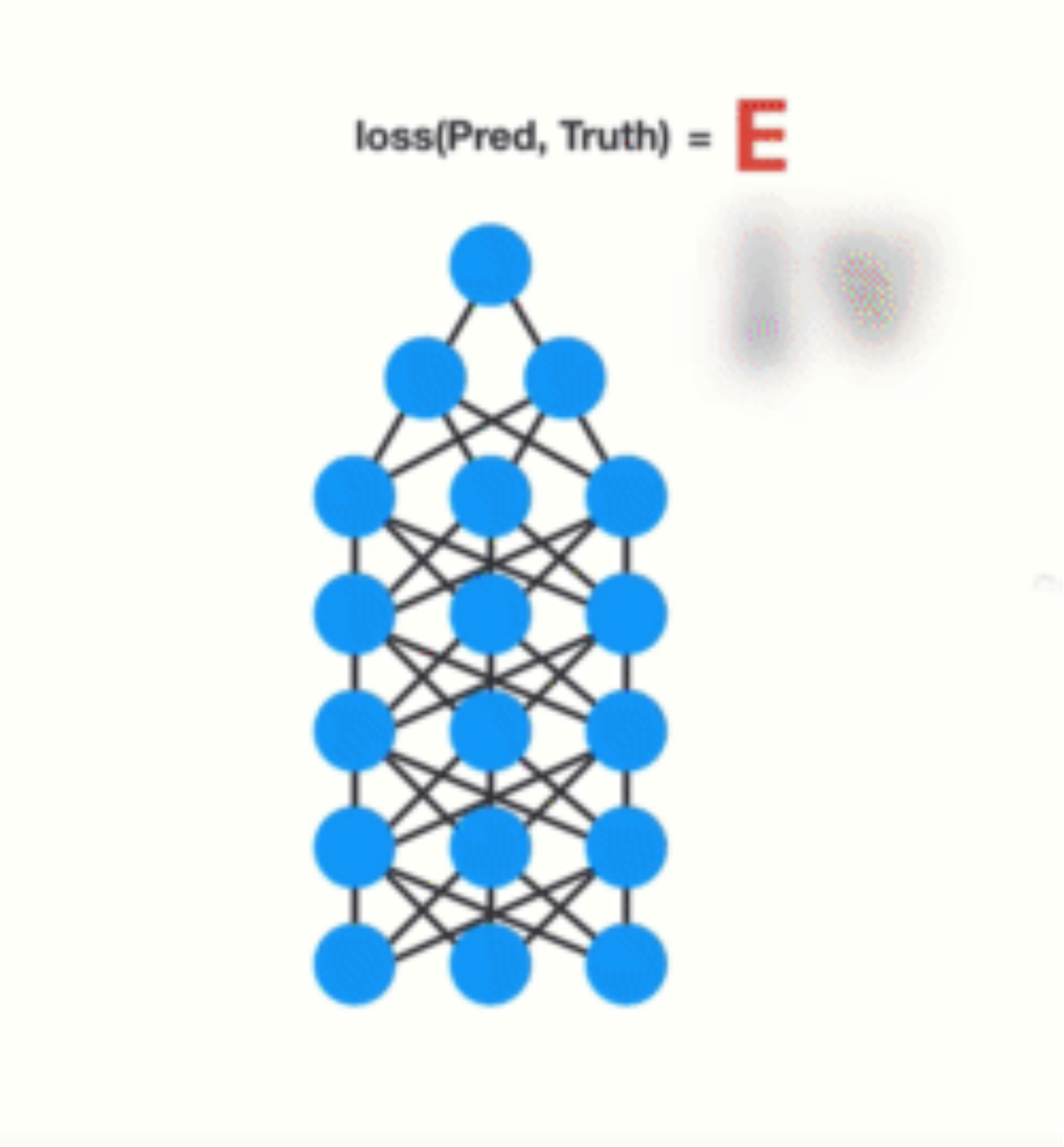
CNNs

Announcements

- Environment
  - I have yet to add "common issues" section for windows.
  - Halfway through updating cluster for more speed.
  - Docker image 1.3
- Assignment 2 set on 9th of April, due 23rd of April.

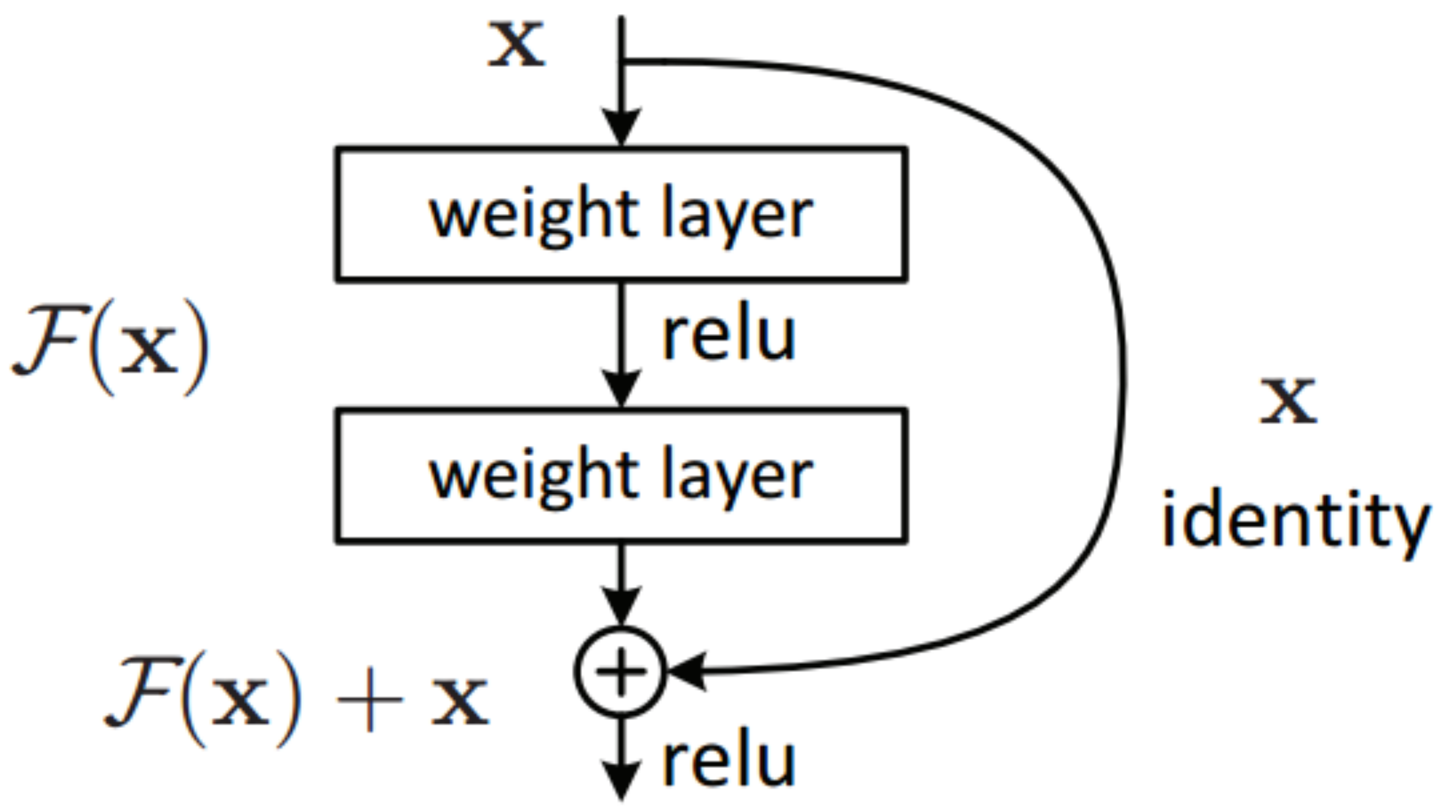
Recap / Questions?

- Training RNNs is hard
- **Vanishing gradient** problem.



Recap / Questions?

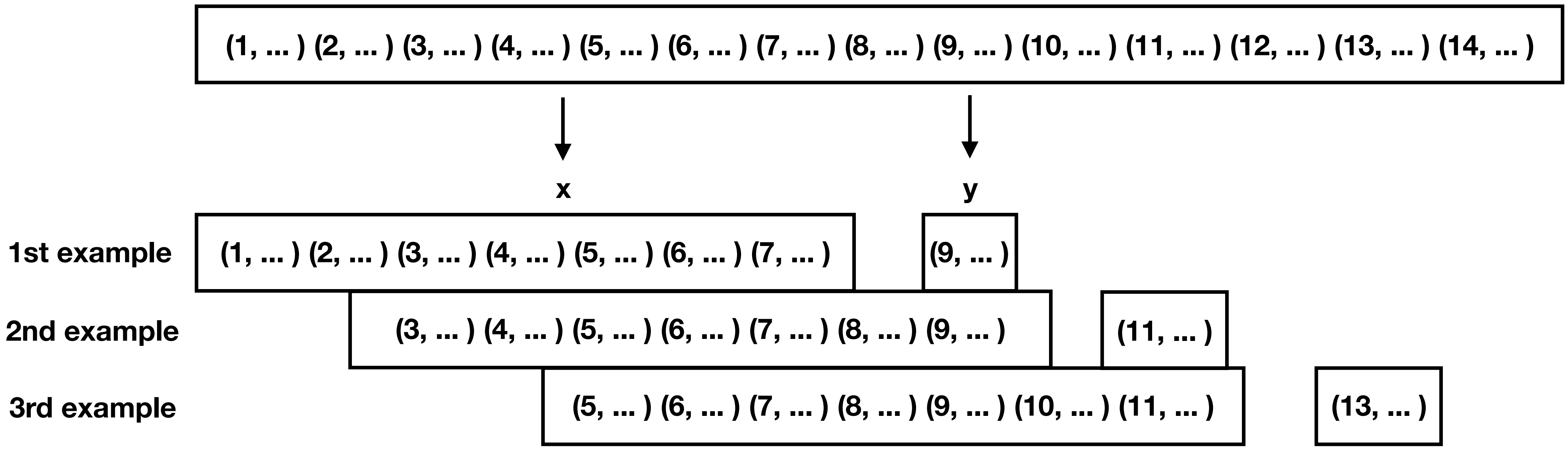
- **GRU** and **LSTM** solve that problem decently.
- Solution was **residual connections**.



Recap / Questions?

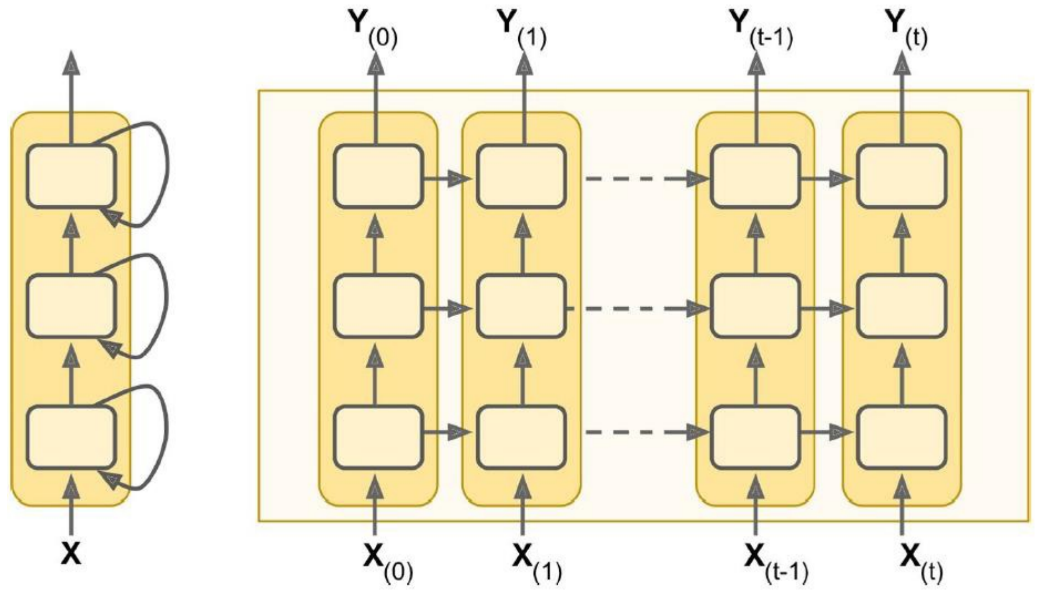
- Generating sequences

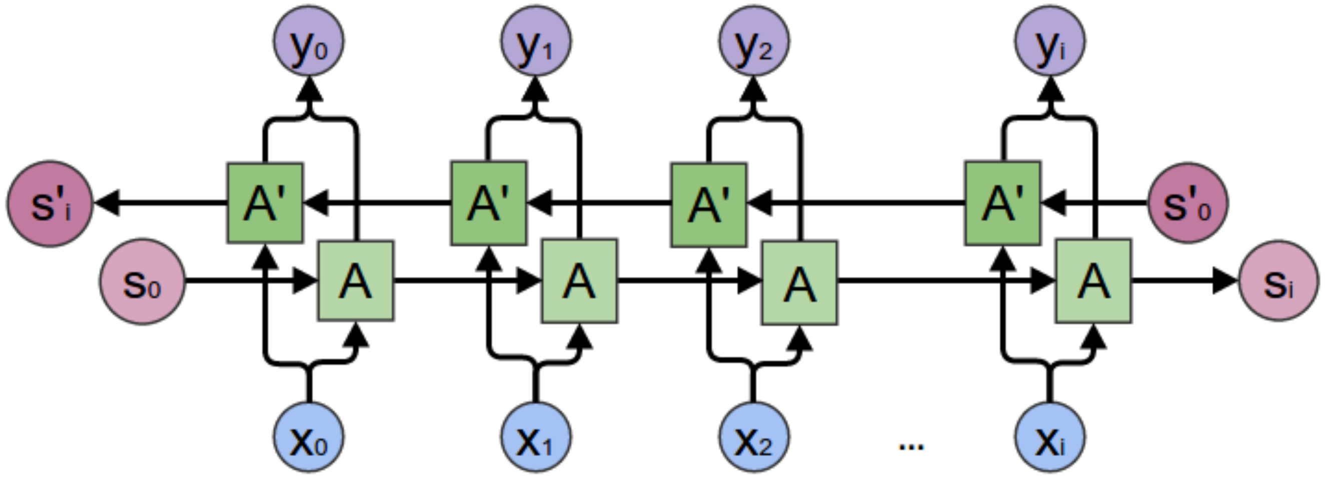
Shift approach, using shift = 2, sequence length = 7, target shift = 1



Recap / Questions?

- Improving RNNs using L1/L2 regularisation and dropout.
- Stacking RNNs
- Stateful RNNs
- Bi-directional RNNs





Source: https://www.simplilearn.com/recurrent-neural-networks-tutorial

SURF

Overview

Today we will cover

Topic: CNNs.

- Image data and signal processing.
- Convolutions.
- Convolutional layers.

Notebook: Simple CNN using convolutions on image data.

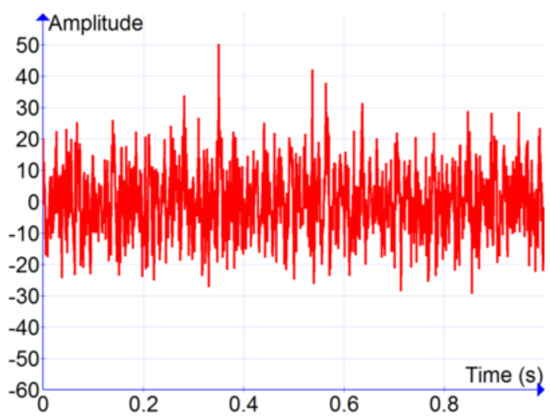
Topic: CNN architectures

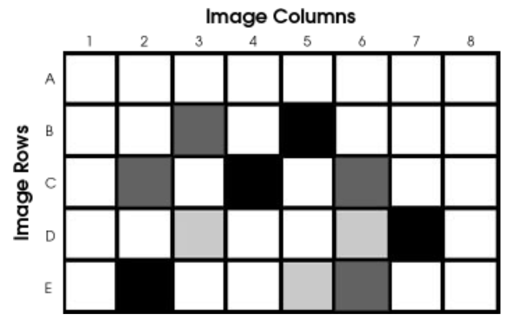
- Max/Avg pool.
- LeNet
- ResNet
- Transfer learning

Notebook: Using a pretrained ResNet50 on image data.

Image data

- Today, a lot about "convolutions" on an image.
- These approaches work well on all data in which the data-points have a "**closeness**" relationship to each other.
    - Sound data
    - Image data
    - Timeseries
- Data can be 1D, 2D, ...
- We talk about picture data.

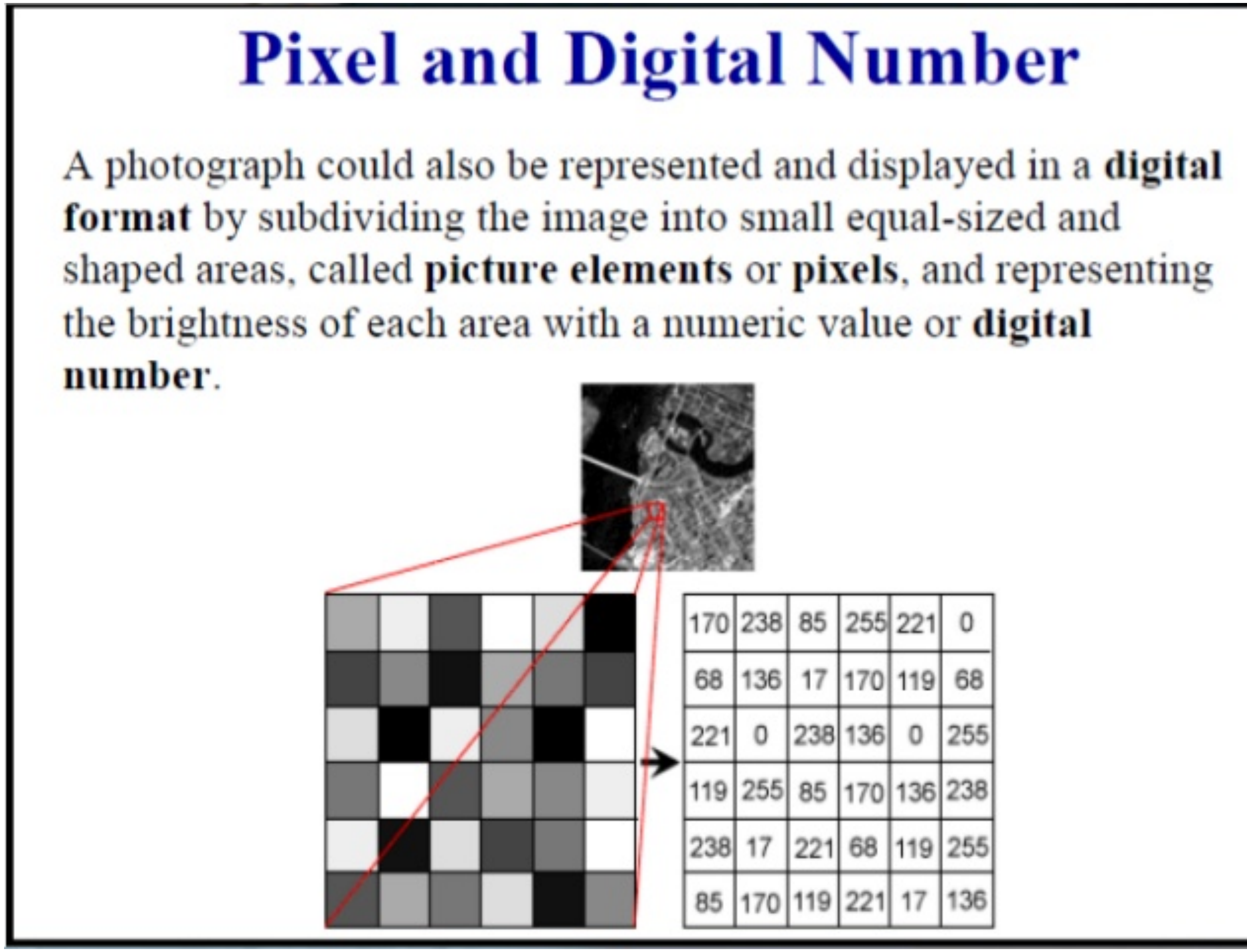




Source: https://en.wikipedia.org/wiki/Signal_processing
& https://earthobservatory.nasa.gov/features/RemoteSensing/remote_06.php

Image data

- An image = Matrix
- Gray = 256 colors (1 byte)
- White = 255
- Black = 0

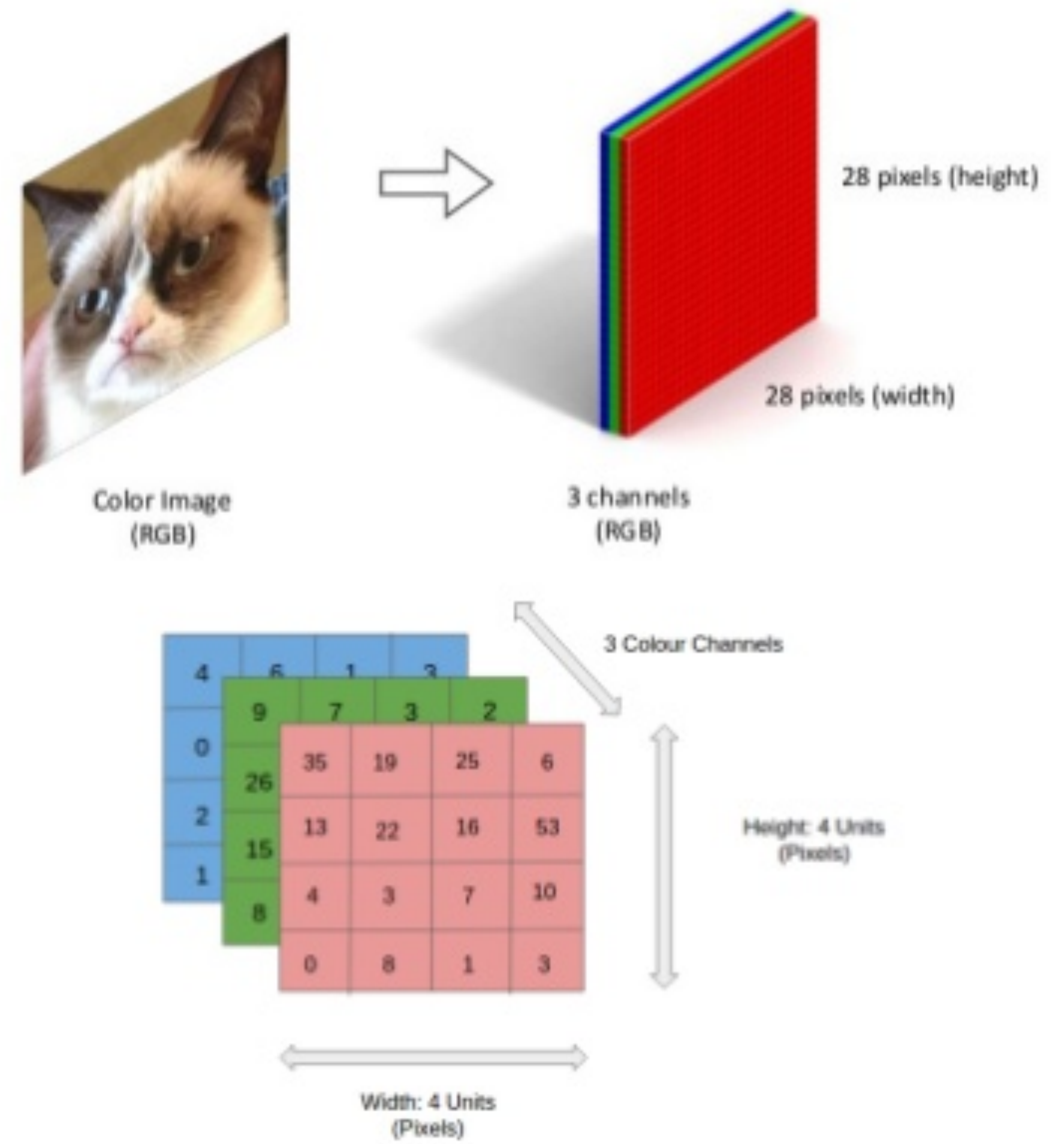


170	238	85	255	221	0
68	136	17	170	119	68
221	0	238	136	0	255
119	255	85	170	136	238
238	17	221	68	119	255
85	170	119	221	17	136

Image data

color image is 3rd-order tensor

- With colors = 3D
- 3 color channels
- Instead of just storing single value between 0-255 we store 3 such values.

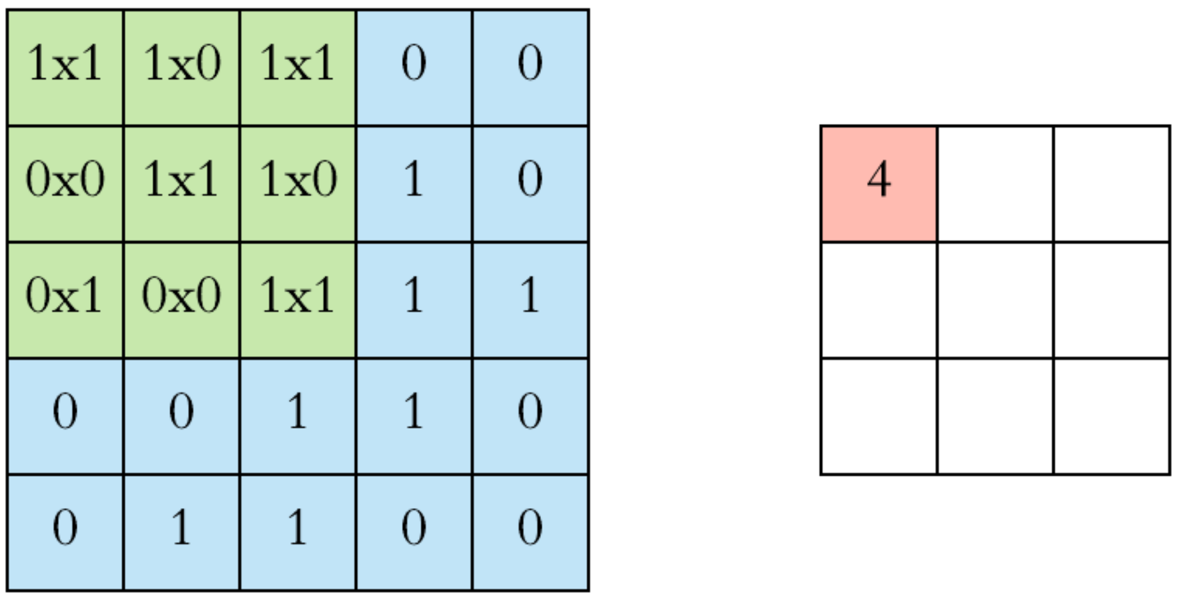


Signal processing

- **Signal processing** deals with pattern detection in such data. Very generically.
    - They need to **represent the data** cleverly when transmitting (streaming online, bluetooth, ...) so no/little information is lost.
    - Deep learning is about learning good representations.
- The main tool borrowed from signal processing is the concept of **convolutions**, or **filters** which are applied repeatedly to the signal in attempts to find a specific pattern.

Convolutions

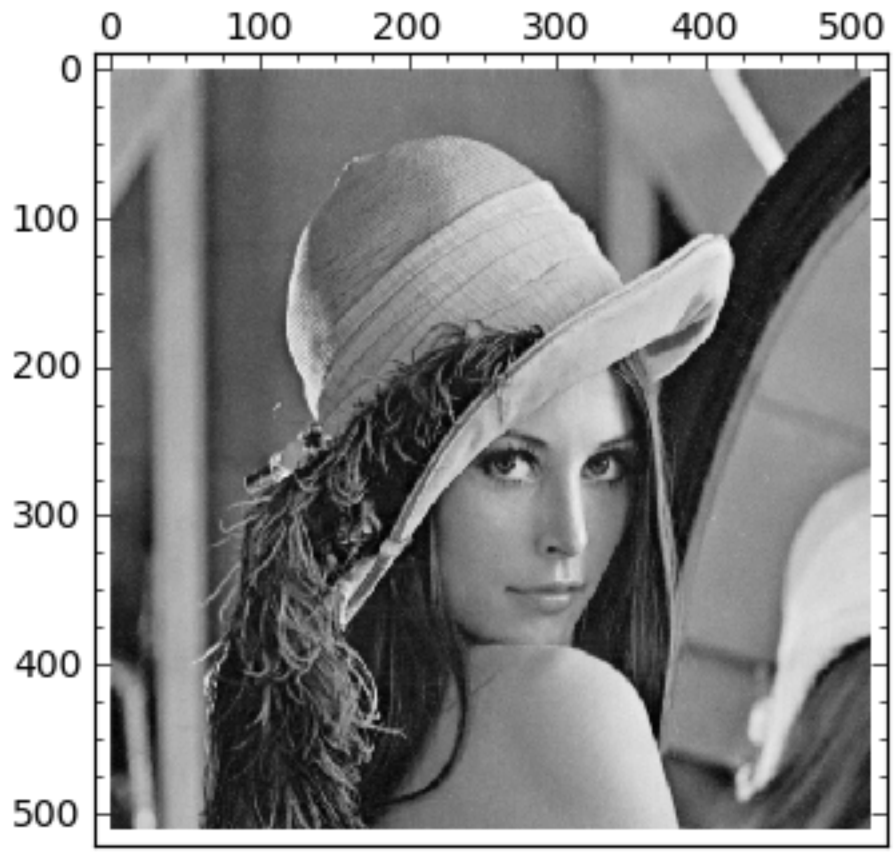
- We can think of convolutions as small filters on the image.
- They process parts of the image and "fire" when they detect the filter pattern.

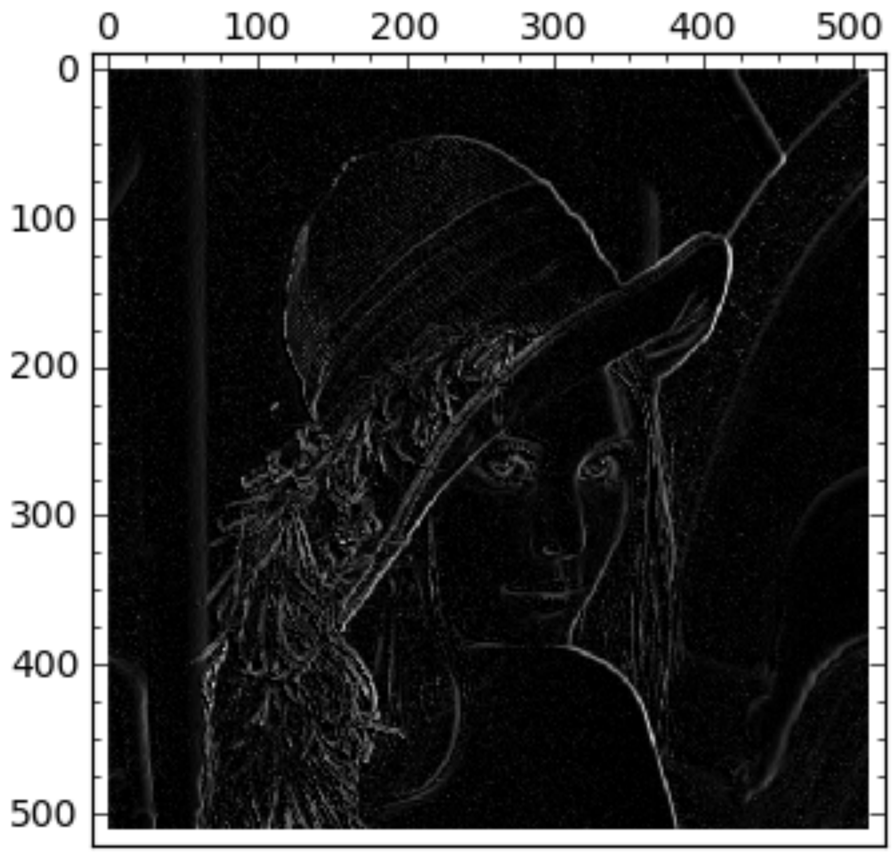


Source: https://medium.freecodecamp.org/an-intuitive-guide-to-convolutional-neural-networks-260c2de0a050

Convolutions

- Applying a filter to an image.

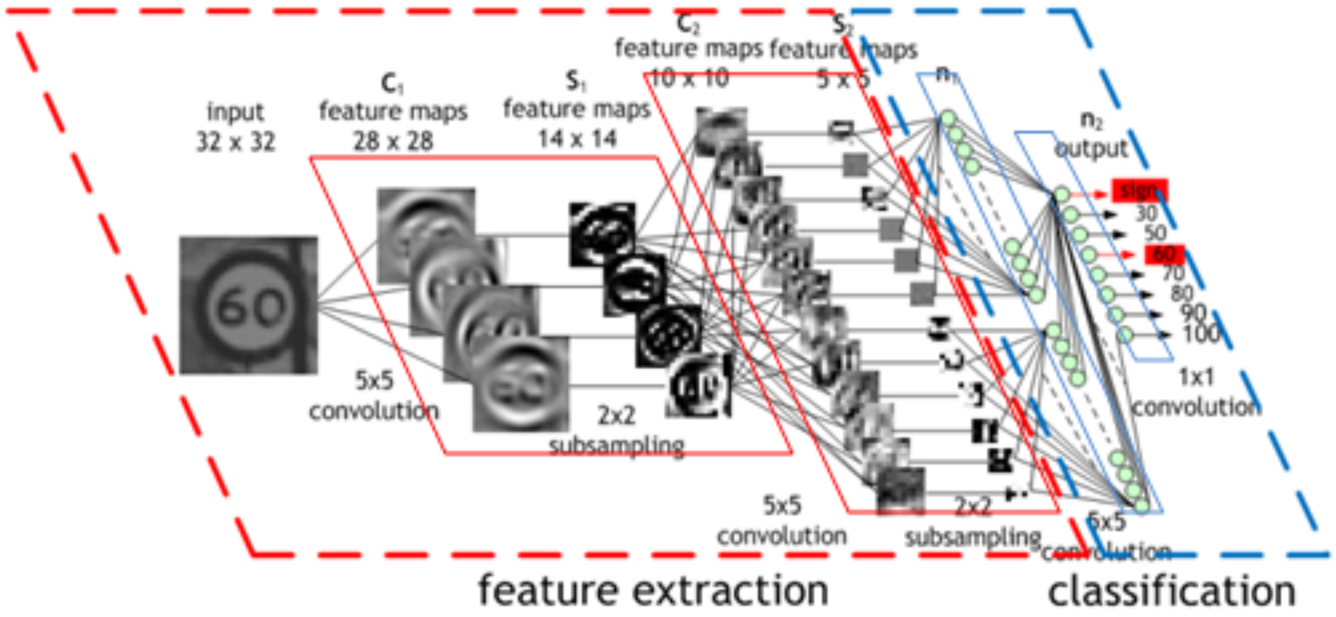




Source: http://gearons.org/2018-02-02-tLSTM/

Convolutions

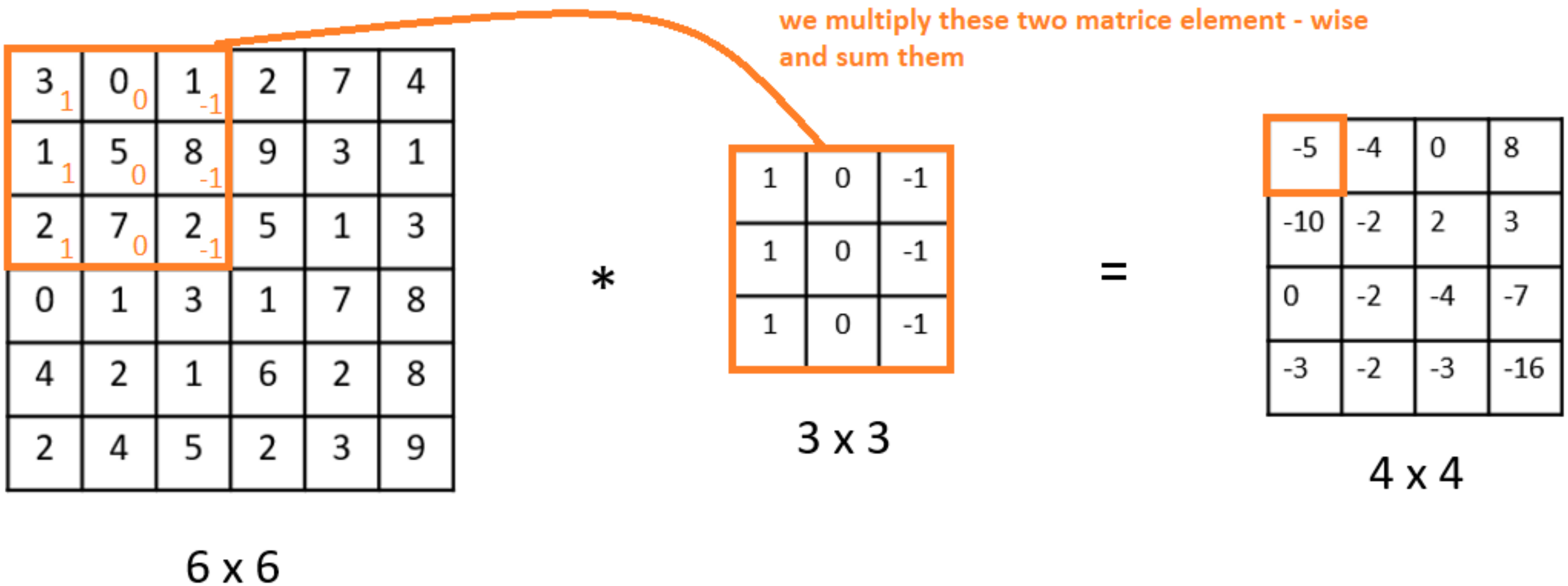
- We then applying a convolution upon a convolution.
- Allows us to create filters from many other filters.
- Then we can detect complex patterns over a larger area.



Source: https://developer.nvidia.com/discover/convolutional-neural-network

Convolutions

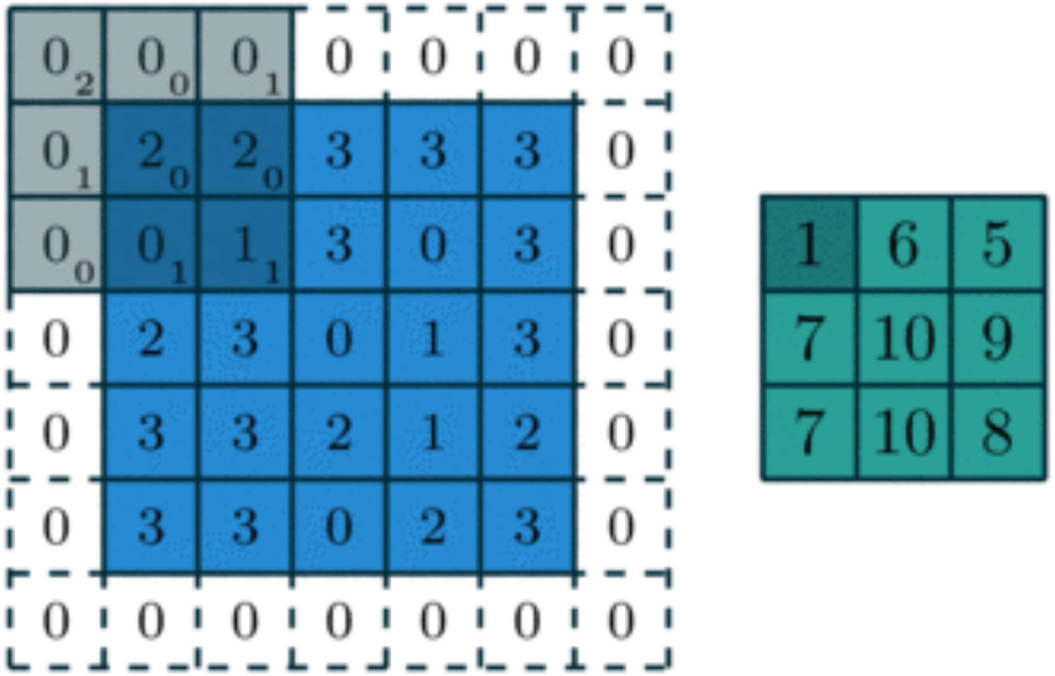
Technicalities



Source: https://developer.nvidia.com/discover/convolutional-neural-network

Convolutions

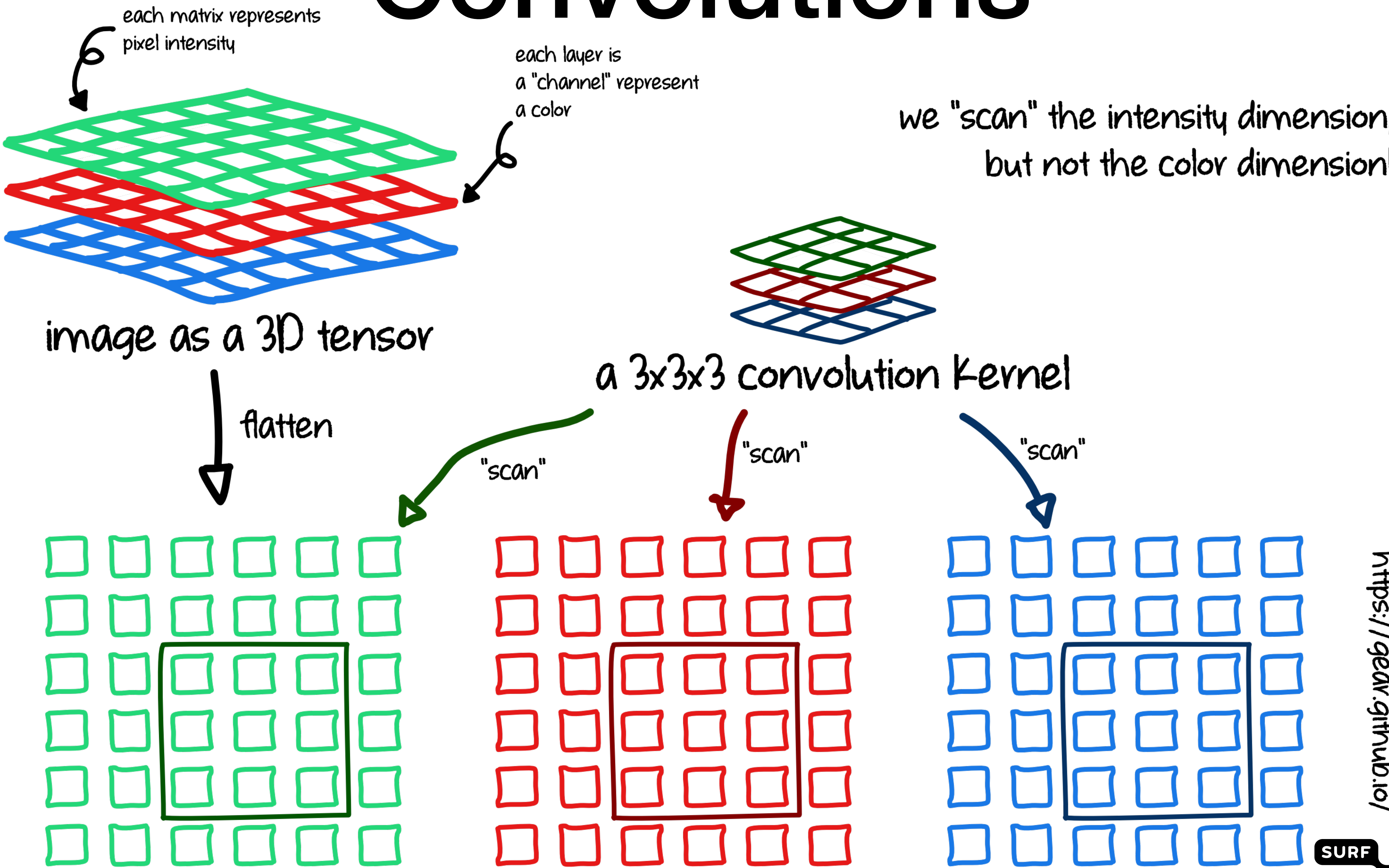
- We apply a **kernel** with dimensions (3, 3).
- Using **stride** = 2.
- **Padding** = 1, to increase output size
- **Same-padding** = Output is same size as input
- Otherwise **valid-padding**.
- Apply activation function to output.
- We try to **learn the filters**! The filters are the parameters/weights.




```
layer_conv_2d(filters = 1,
    kernel_size = c(3, 3),
    stride = 2,
    padding = "valid",
    activation = "",
    kernel_regularizer = "",
    input_shape = c(5, 5))
```

Source: http://deeplearning.net/software/theano_versions/dev/tutorial/conv_arithmetic.html

Convolutions

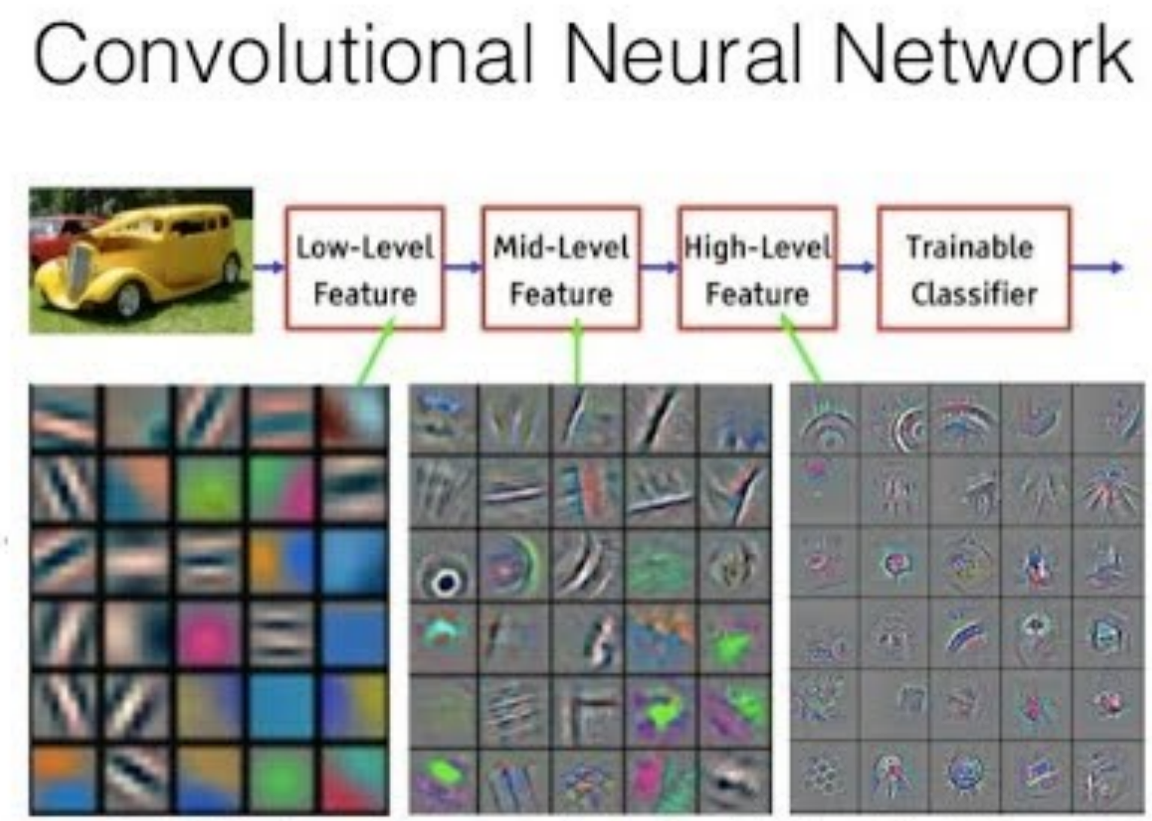


https://gear.github.io/

Convolutions

- We then create many filters/ kernels in each layer.

```
layer_conv_2d(filters = 32,
    kernel_size = c(3, 3),
    stride = 2,
    padding = "valid",
    activation = "",
    kernel_regularizer = "",
    input_shape = c(5, 5))
```



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

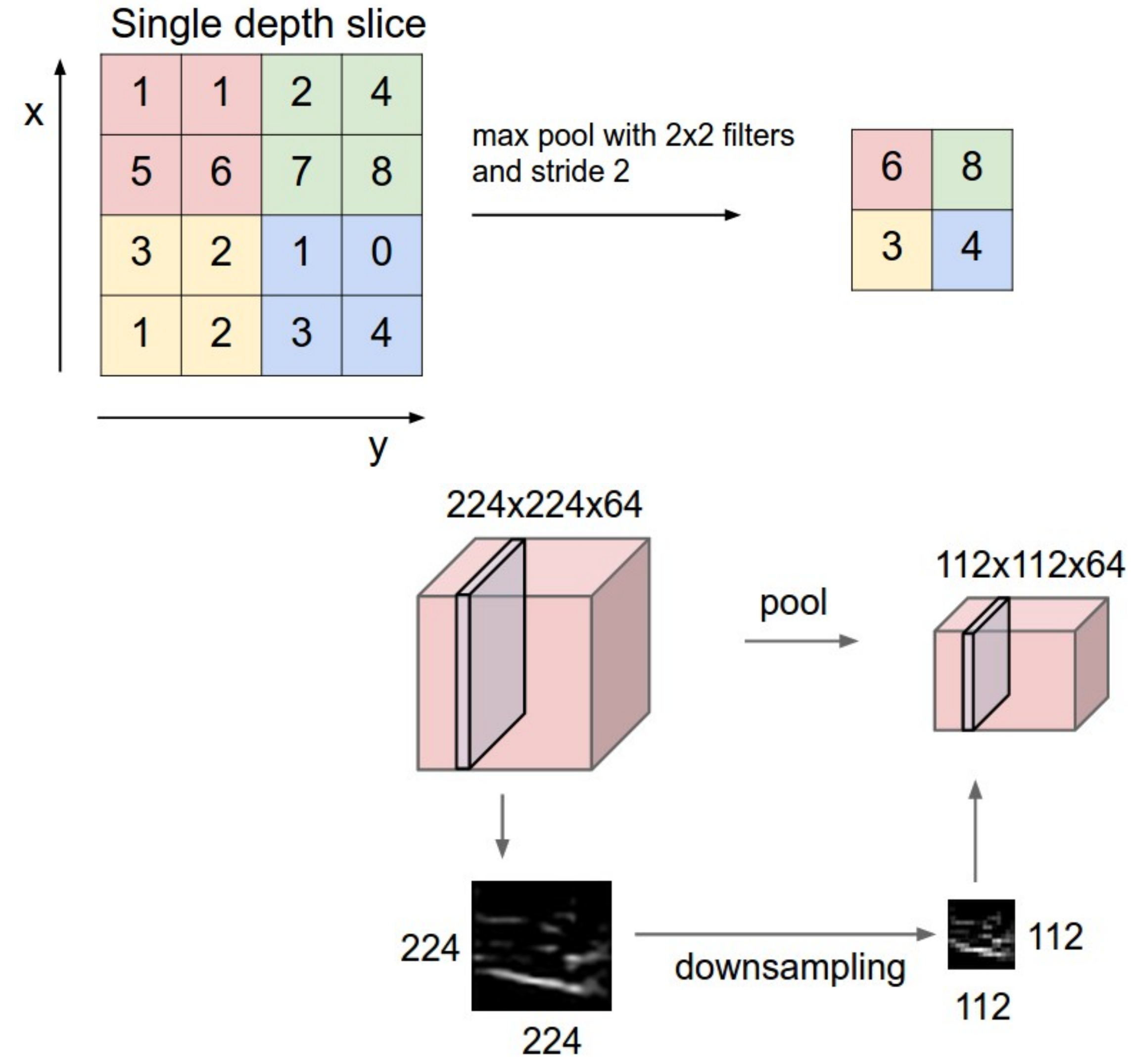
Hands-on

Go to https://dba.projects.sda.surfsara.nl/

Notebook: `06a-cnns.ipynb`

CNNs

Pooling layers



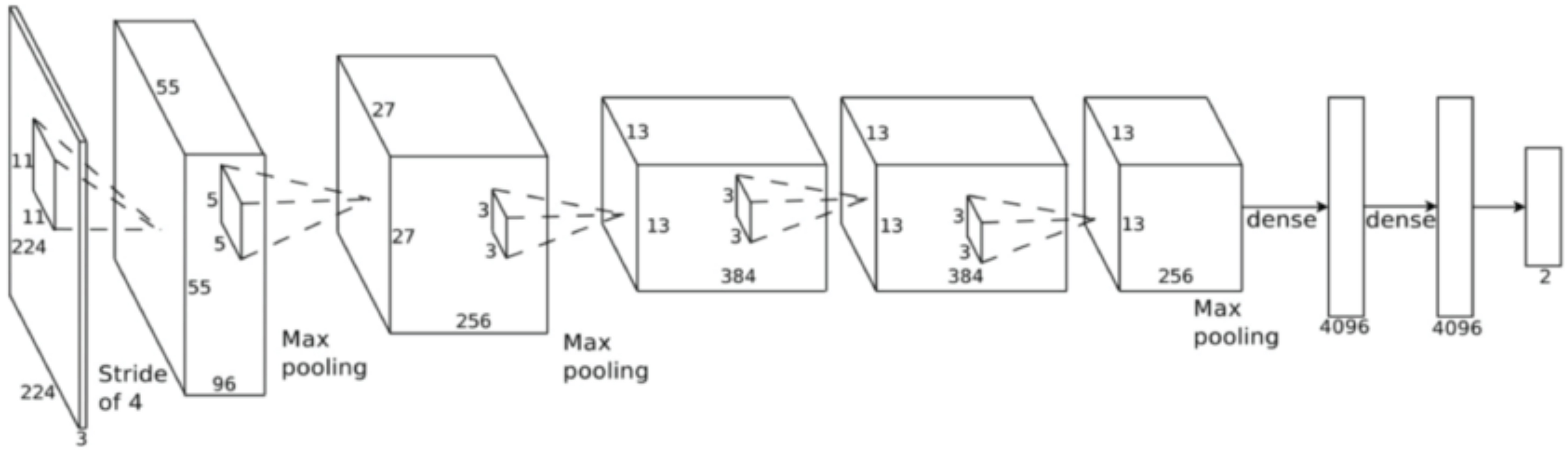
- Our filter maps are sometimes very large, so we make them **smaller using pooling**.
- **Max pooling**, take the max.
- **Average pooling**, take the average.
- Applied after convolutions

Source: http://deeplearning.net/software/theano_versions/dev/tutorial/conv_arithmetic.html

CNNs

Architectures - AlexNet

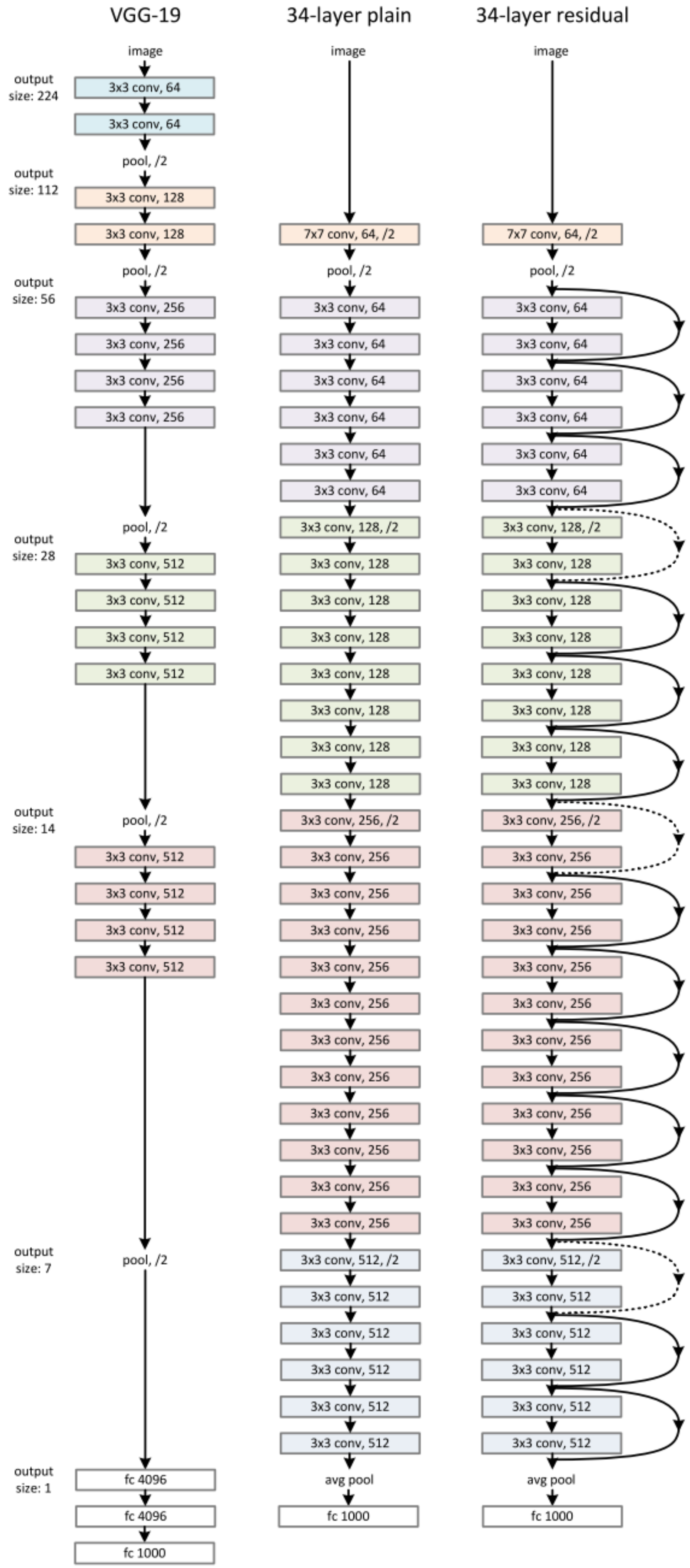
- By chaining different combinations of them we can create Convolutional Neural Networks (CNNs).



Source: https://www.jeremyjordan.me/convnet-architectures/

CNNs

Architectures - ResNet (2015)

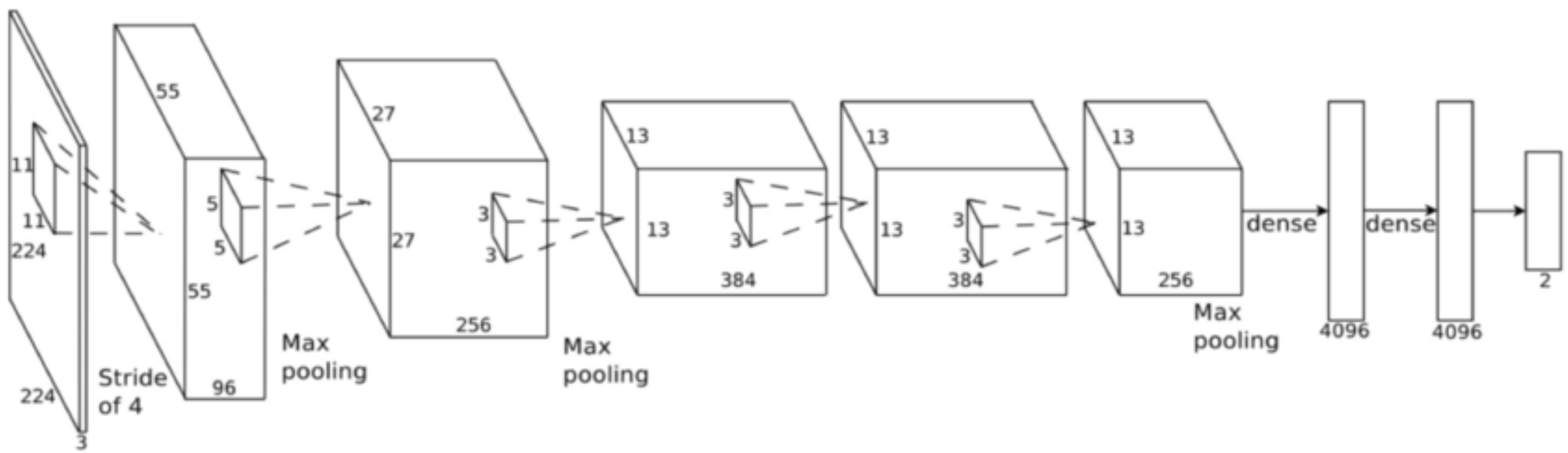


Source: https://arxiv.org/abs/1512.03385

CNNs

Pretrained

- Should we develop these architectures ourselves?
- No, we used them, pre-trained
- This is called **transfer learning**.



Source: https://www.jeremyjordan.me/convnet-architectures/

Transfer learning

- We use the fact that someone trained the model on a very large dataset, using **more data and compute** than we can (usually) hope to achieve.

- The last part is always specific to the task at hand, we want the "general" part, so we do not use the whole network.

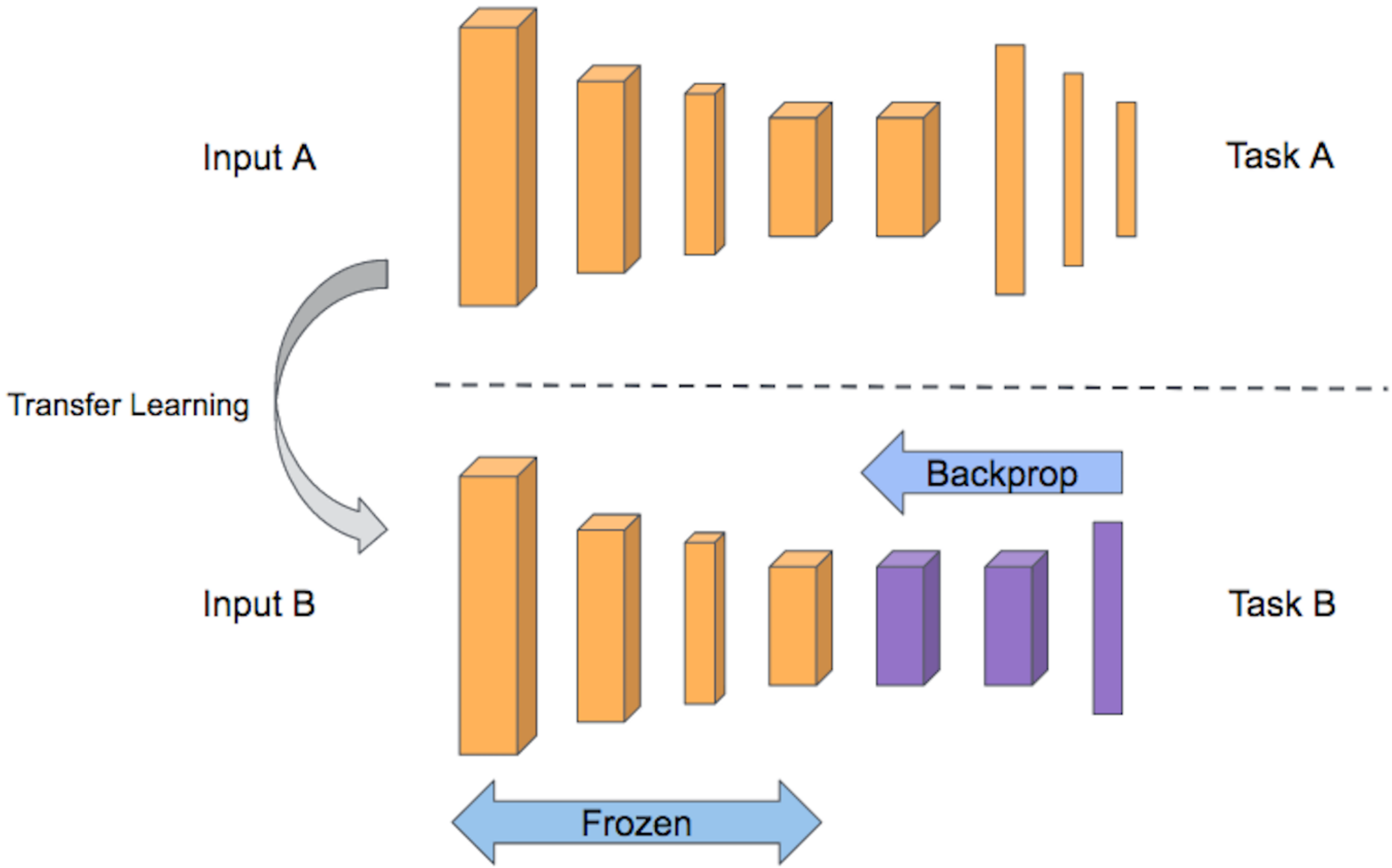
- We **do not train the network**, just the part we add.

- Our hope is that the network has learnt some generally **good representations** we can use.

- Not training the weights of the network is called **freezing the weights**.

Source: https://www.jeremyjordan.me/convnet-architectures/

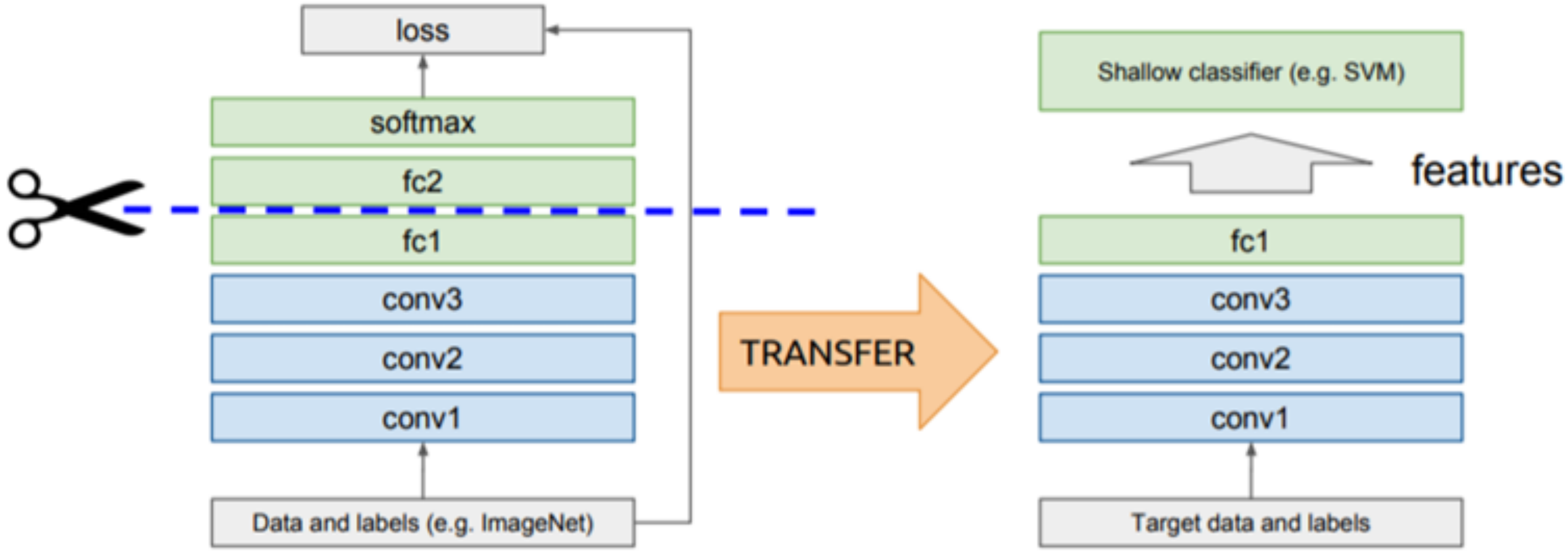
Transfer learning



Transfer learning

Idea: use outputs of one or more layers of a network trained on a different task as generic feature detectors. Train a new shallow model on these features.

Assumes that $D_S = D_T$



Transfer learning

Caveats

- We need to **download** the network and weights, which can be large (still 1000 times faster than training ourselves).

- We use a **smaller learning rate**.

- To increase speed, many people preprocess the input through the static network and **save the representations to disk** and then train a new network separately.

Source: https://www.jeremyjordan.me/convnet-architectures/

Hands-on

Go to https://dba.projects.sda.surfsara.nl/

Notebook: `06b-cnns-transfer.ipynb`

Wrap-up at 12:20 / 16:20

Transfer learning

- You also see transfer learning in RNNs, especially with word embeddings.

Source: https://www.jeremyjordan.me/convnet-architectures/

Summary

Topic: CNNs.

- Image data and signal processing.
- Convolutions.
- Convolutional layers.

Notebook: Simple CNN using convolutions on image data.

Topic: CNN architectures

- Max/Avg pool.
- LeNet
- ResNet
- Transfer learning

Notebook: Using a pretrained ResNet on image data.