OEAW AI SUMMER SCHOOL

DEEP LEARNING III

Convolutional Neural Networks

Johannes Brandstetter, Michael Widrich
Institute for Machine Learning

JKU
Institute for
Machine Learning

Lecture III: Convolutional Neural Networks

Basics of Convolutional Neural Networks (ConvNets/CNNs)

Neural Nets for Image Recognition

- ImageNet Large Scale Visual Recognition Competition (ILSVRC)
  - 1.2M images, 1 000 different classes
  - Yearly challenge to find the ‘state of the art” in image recognition
- ILSVCR dominated by ConvNets:
  - ILSVRC 2012: Won by the only CNN-based solution
  - ILSVRC 2013: Best 5 participants were all CNNs (9 of the top 10 were CNNs)
  - ILSVRC 2014: Everyone uses CNNs
- CNNs are useful whenever there is “local structure” in the data:
  - Pixel data
  - Audio data
  - Voxel data
  - ...

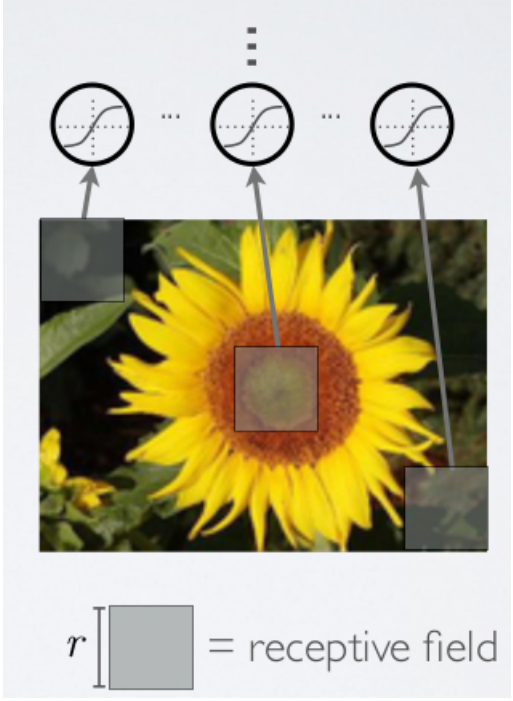
IMAGENET

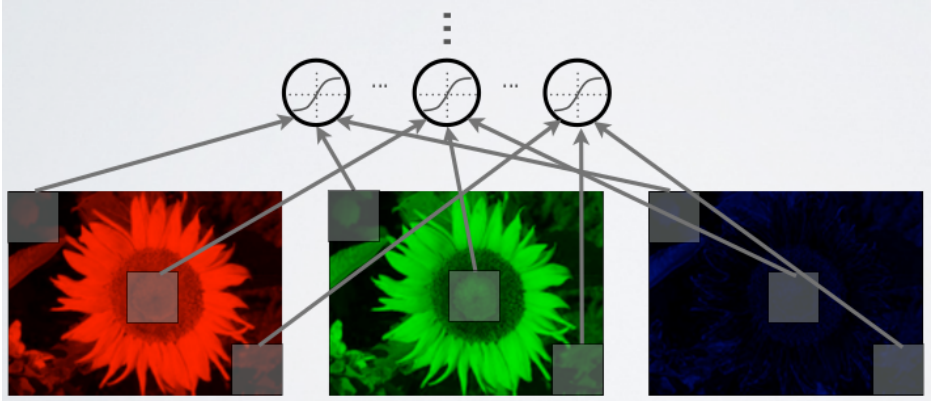
Basic considerations for ImageNet data

- Images are extremely high-dimensional:
  - 250 x 250 pixels * 3 color channels = 200k input dimensions
- Pixels that are near each other are highly correlated.
- Often invariances to certain variations are desired (e.g. translations).

Local Connectivity

- Instead of fully connected units, let each unit only see a small image "patch".
- Number of parameters is reduced.



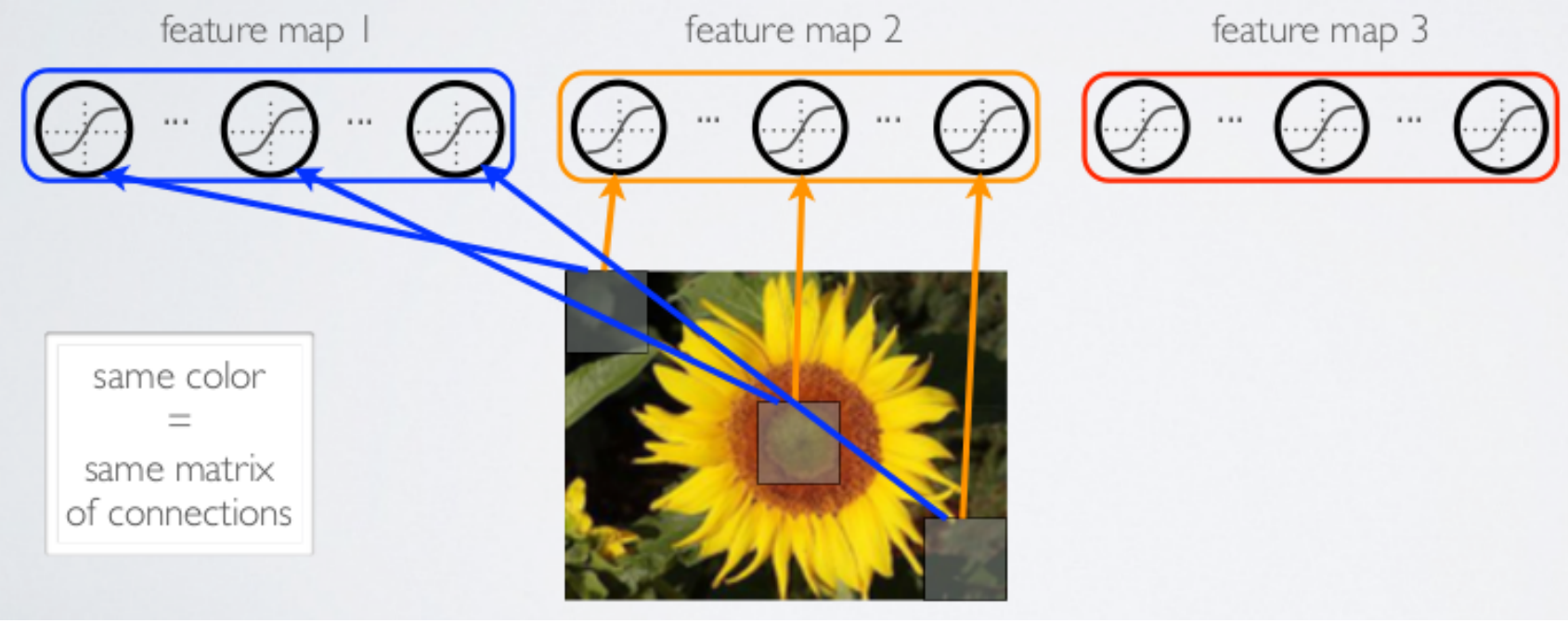


Weight Sharing

- Image patches can be decomposed into basic image patches.

Local Connectivity

- Same basic patches appear on all positions of the image.
- Typically edge-detectors, corners, ...
- Idea to use same receptive fields on all image positions:
  - Sharing parameters between receptive fields



[Images by Hugo Larochelle]

Convolution

- A receptive field is applied to each position (i, j) of the image to obtain a value z_{ij}:

$$z_{ij} = \sum_{a=0}^{a=r-1} \sum_{b=0}^{b=r-1} W_{ab} x_{i+a,j+b}$$

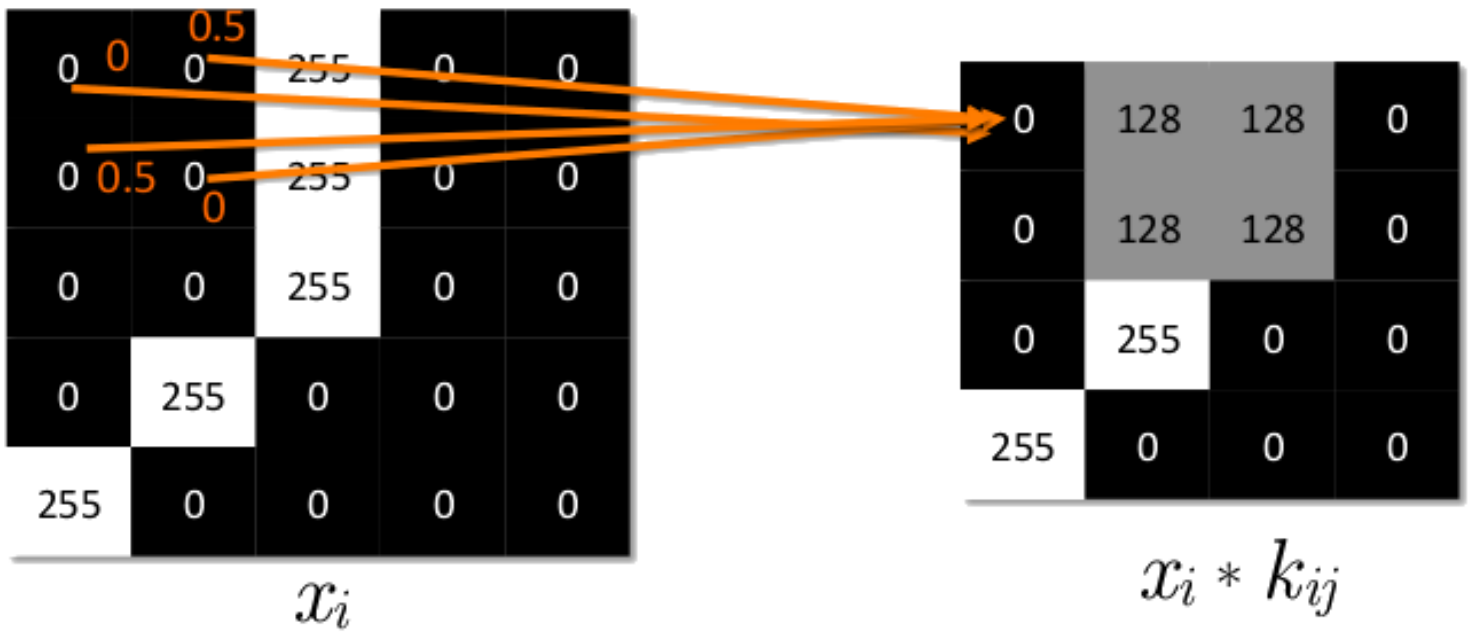
- This is called a (discrete) convolution $\mathbf{X} * \mathbf{W}$.
- We apply an activation function to z_{ij} afterwards.

Example convolution

Topics: discrete convolution

Simple illustration: $x_i * k_{ij}$ where $k_{ij} =$

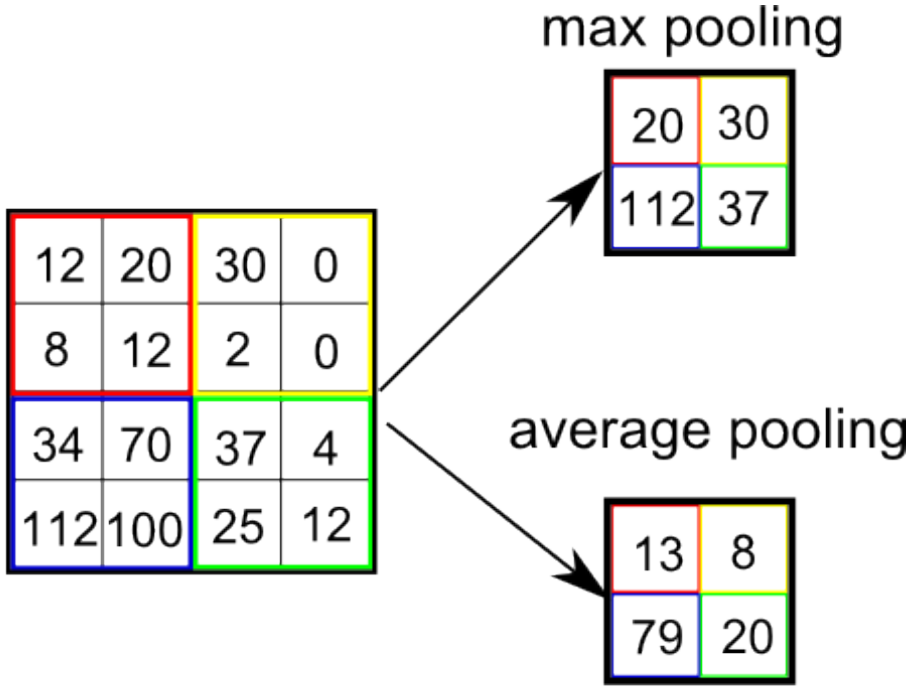
0	0.5
0.5	0



[Images by Hugo Larochelle]

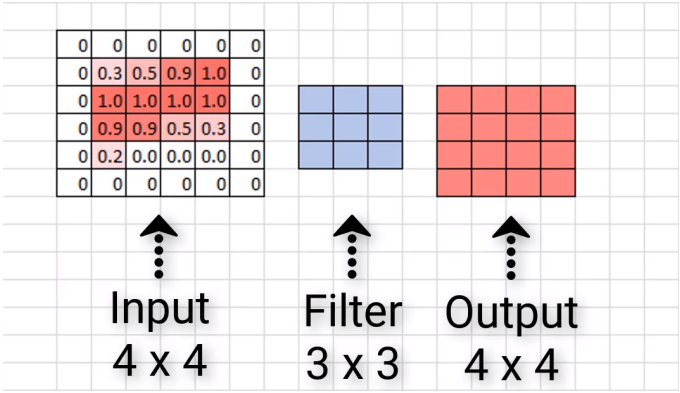
Subsampling ("Pooling")

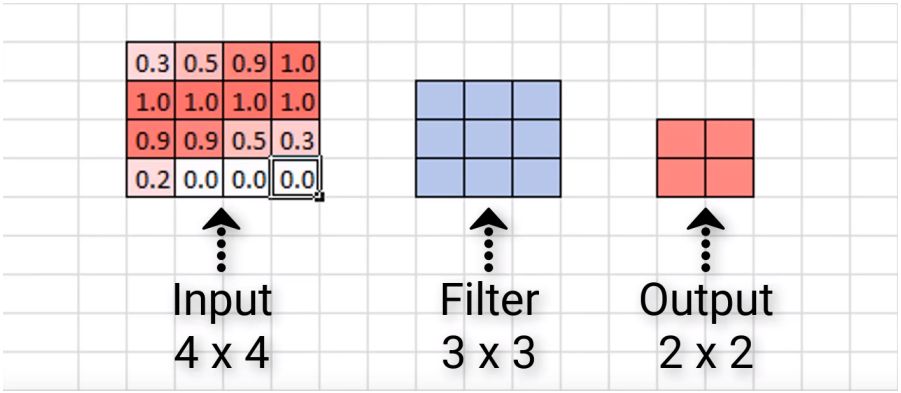
- We can subsample an image to produce a smaller image.
- We can do the same to the result of a convolution.
- This is called **pooling**.
- There are different ways to perform pooling. Most popular:
  - Average Pooling: take the average of all NxN numbers
  - Max Pooling: take the maximum of the numbers



Striding and padding

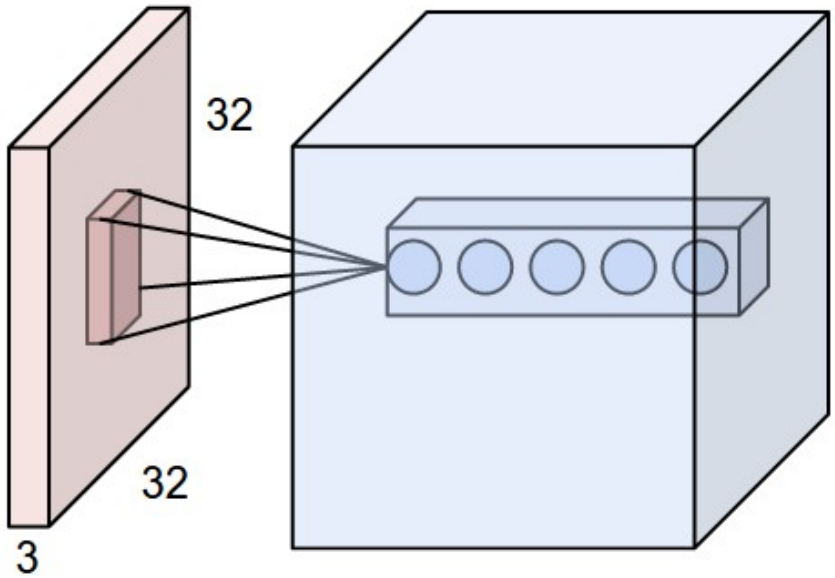
- Striding controls how much the filters are moved. The smaller the stride the more the receptive filters overlap.
- Padding provides control to the output volume by adding zeros.





Depth in CNNs

- Depth is the third dimension in a convolutional layers.
- When convolving a layer of size $x \times x \times \boldsymbol{n}$ with $\boldsymbol{m}$ kernels of the size $x' \times x' \times \boldsymbol{n}$, you end up with an output of size $x'' \times x'' \times \boldsymbol{m}$.
- Don't worry:
  - You will understand in 2 slides
  - Pytorch/Tensorflow does it for you



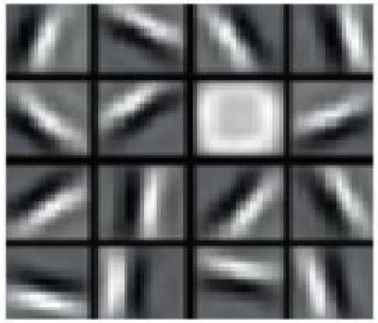
Multiple Levels of Convolutions

A typical CNN architecture has several layers of:

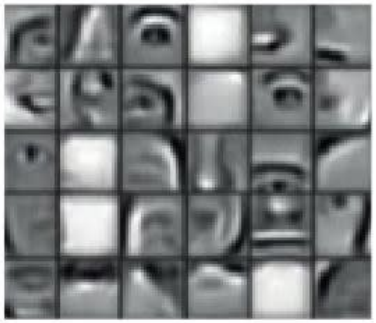
1. Convolution
2. Nonlinearity
3. Pooling (optional)

- Each receptive field produces a new channel/feature map for the next layer.
- Convolution operates on multiple receptive fields and multiple feature maps.
- The complexity of detected features tends to increase layer by layer (first feature map does edge detection, later ones combine it to complex shapes).

Facial Recognition



1. EDGES



2. FEATURES

3. FACES

4. FULL FACE

DeepFace: Facial recognition system used by Facebook since 2015

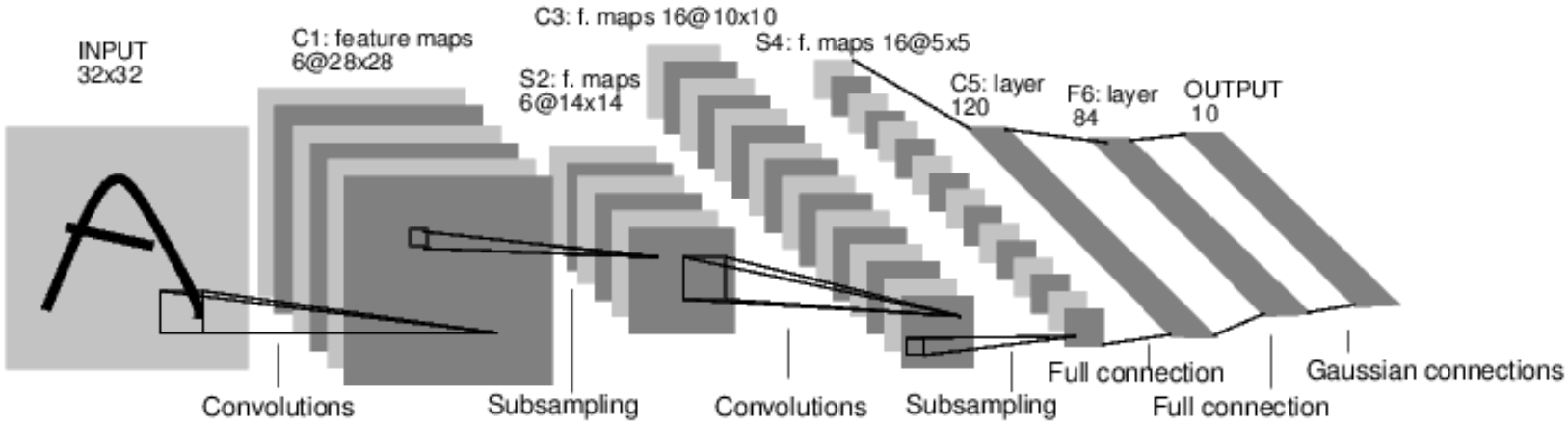
[Source https://www.simplilearn.com/deep-learning-tutorial]

Pytorch implementation CNNs

```
# CNN architecture (two conv layers followed by two fully connected layers)
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        # nn.Conv2d(in_channels, out_channels, kernel_size, stride=1, padding=0)
        self.conv1 = nn.Conv2d(1, 20, 5, 1)
        self.conv2 = nn.Conv2d(20, 50, 5, 1)
        self.fc1 = nn.Linear(4*4*50, 500)
        self.fc2 = nn.Linear(500, 10)

    def forward(self, x):
        #Size changes from (1, 28, 28) to (20, 24, 24)
        x = F.relu(self.conv1(x))
        #Size changes from (20, 24, 24) to (20, 12, 12)
        x = F.max_pool2d(x, 2, 2)
        #Size changes from (20, 12, 12) to (50, 8, 8)
        x = F.relu(self.conv2(x))
        #Size changes from (50, 8, 8) to (50, 4, 4)
        x = F.max_pool2d(x, 2, 2)
        #Size changes from (50, 4, 4) to (50*4*4)
        x = x.view(-1, 4*4*50)
        x = F.relu(self.fc1(x))
        x = self.fc2(x)
        return F.log_softmax(x, dim=1)
```

LeNet



Gradient-based learning applied to document recognition, LeCun, Bottou, Bengio, Haffner, Proceedings of the IEEE,1998

- First famous CNN
- Basic design still valid today

AlexNet

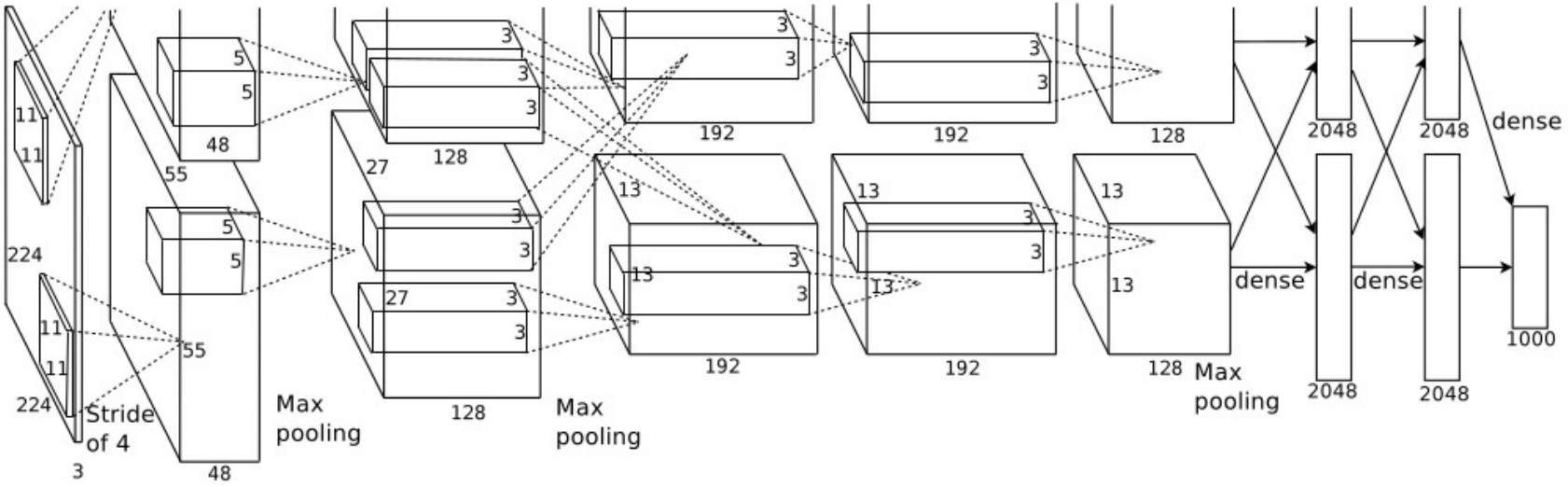


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

- Won ILSVRC 2012 (using ReLUs + Dropout + GPUs) by a landslide
- After Kriezhevsky et al. won ILSVRC 2012, "everyone" started using CNNs for image tasks.

ImageNet Classification with Deep Convolutional Neural Networks, Kriezhevsky, Sutskever, Hinton, NIPS 2012

ResNet

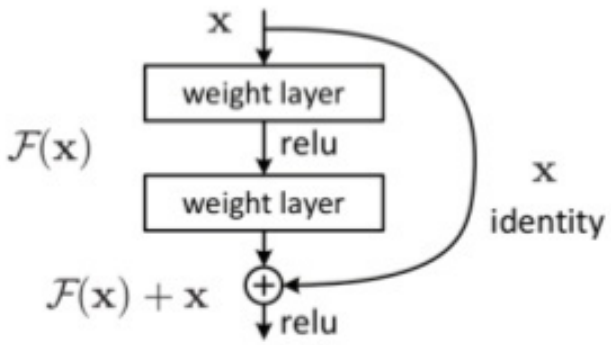


Figure 2. Residual learning: a building block.

- One of the current standard architectures
- Up to 1k layers

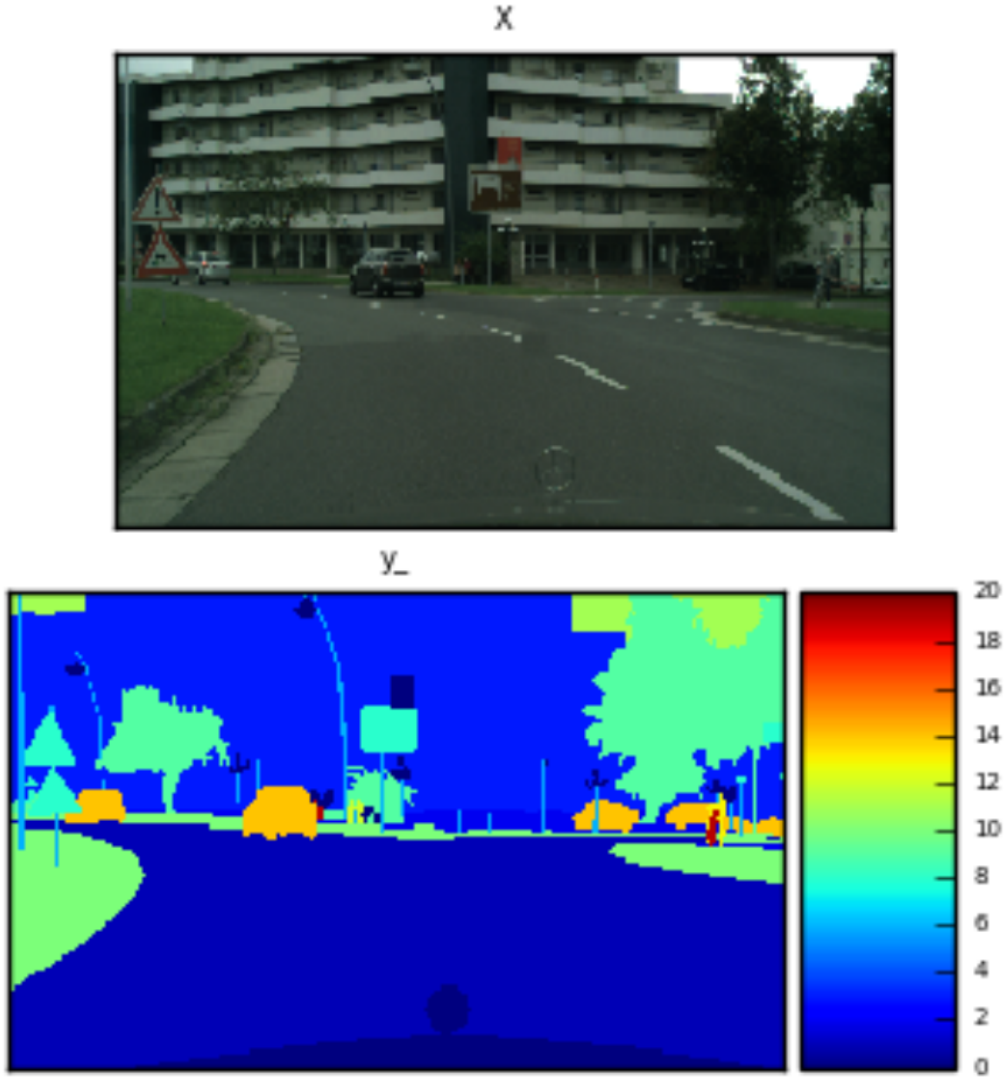
Deep Residual Learning for Image Recognition, He et al., CVPR 2016

Common tricks for image classification

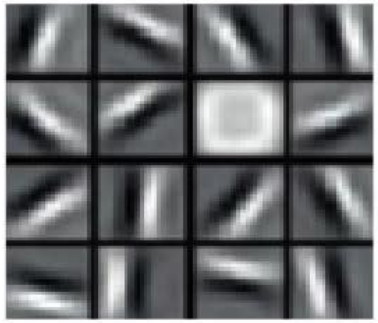
- Dataset augmentation: Create new images from existing ones
  - Mirroring
  - Translation/sharing
  - Scaling
  - Color-channel adaptations
  - Image distortions
  - ...
- Use pretrained nets (Transfer Learning)
- Be aware of overfitting to background/colors/artifacts
- Use striding/pooling to reduce dimensionalty and increase number of filters

Application to Image Data

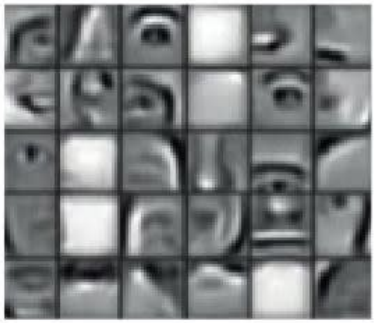
Semantic Segmentation: Self-Driving Cars



Facial Recognition



1. EDGES



2. FEATURES

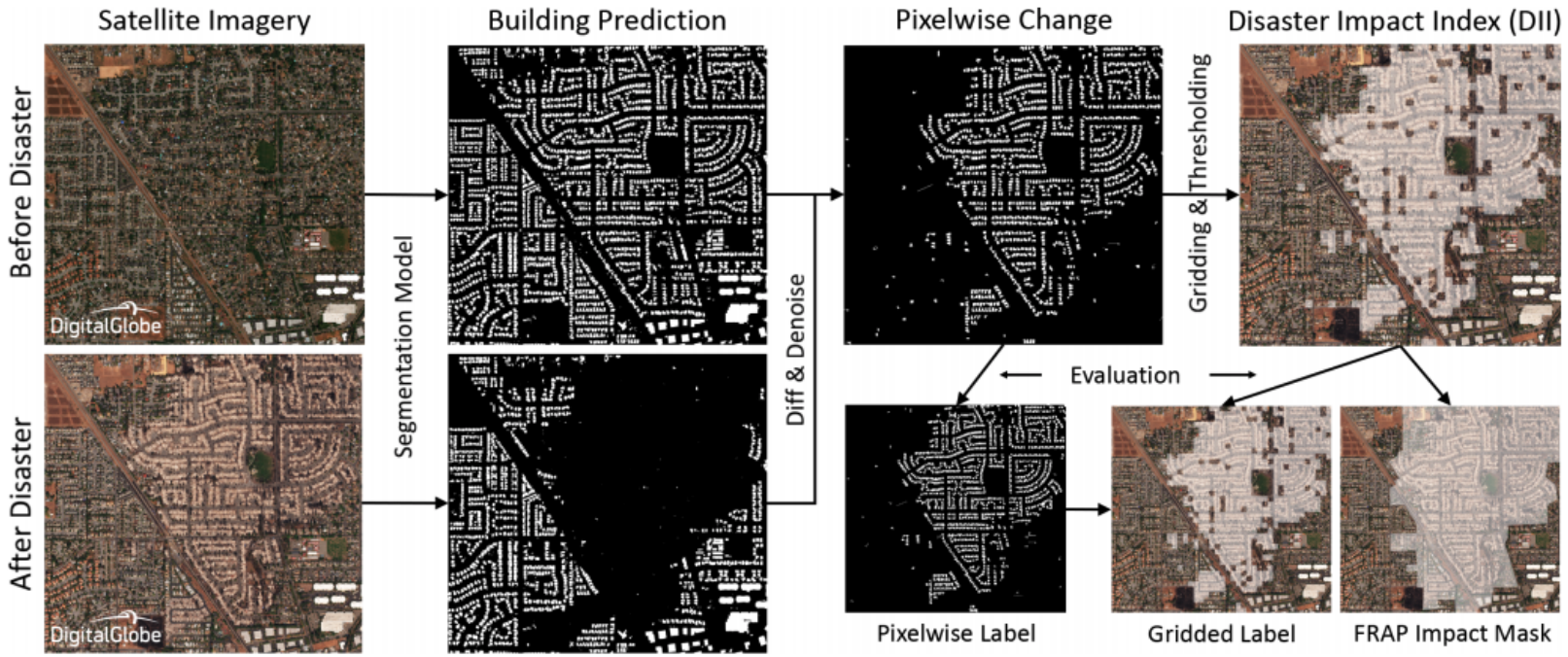
3. FACES

4. FULL FACE

DeepFace: Facial recognition system used by Facebook since 2015

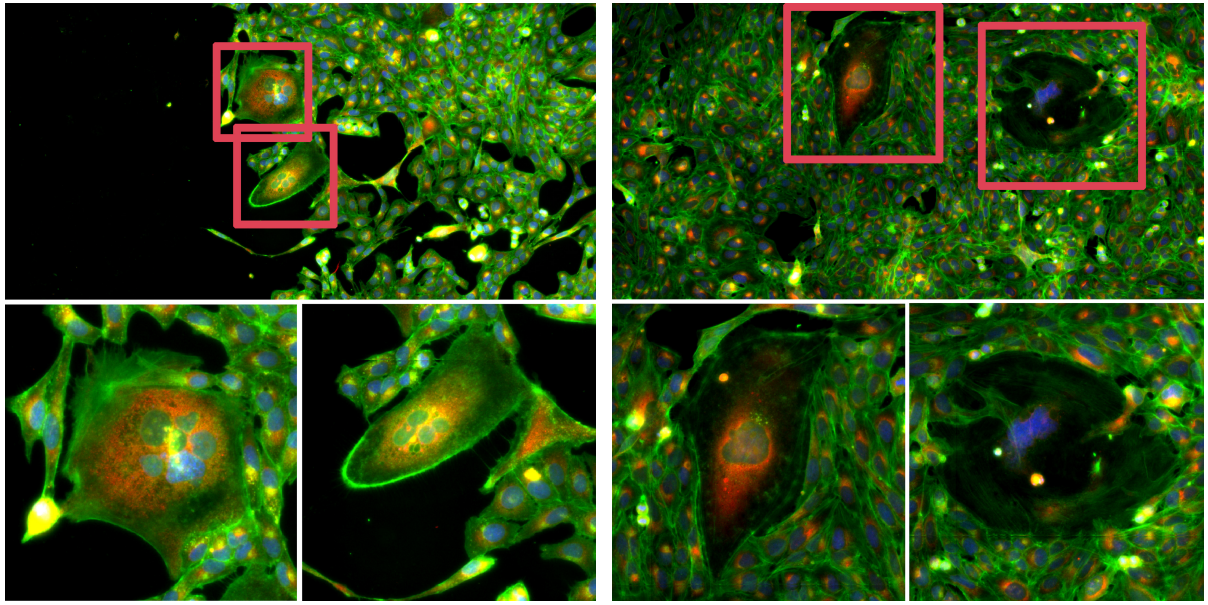
[Source https://www.simplilearn.com/deep-learning-tutorial]

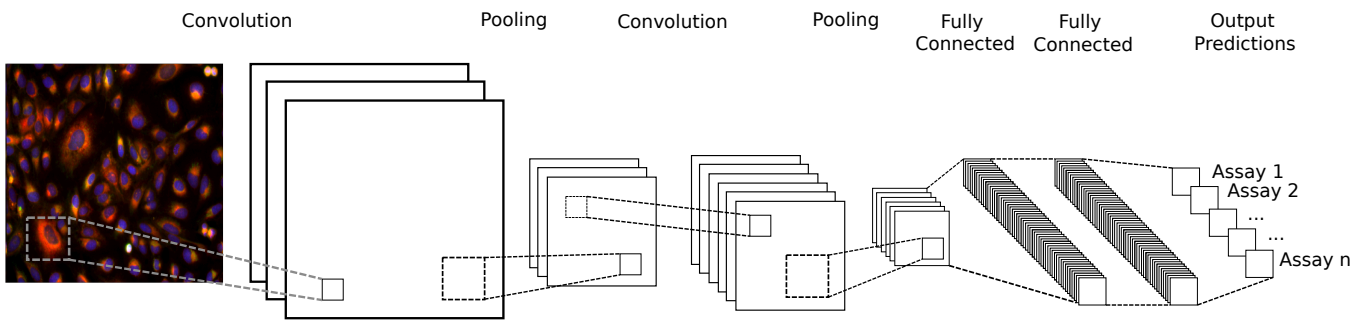
Semantic Segmentation: Satellite Images



Damage estimate after natural disasters [Source: Facebook/CrowdAI]

High-Resolution Image Classification



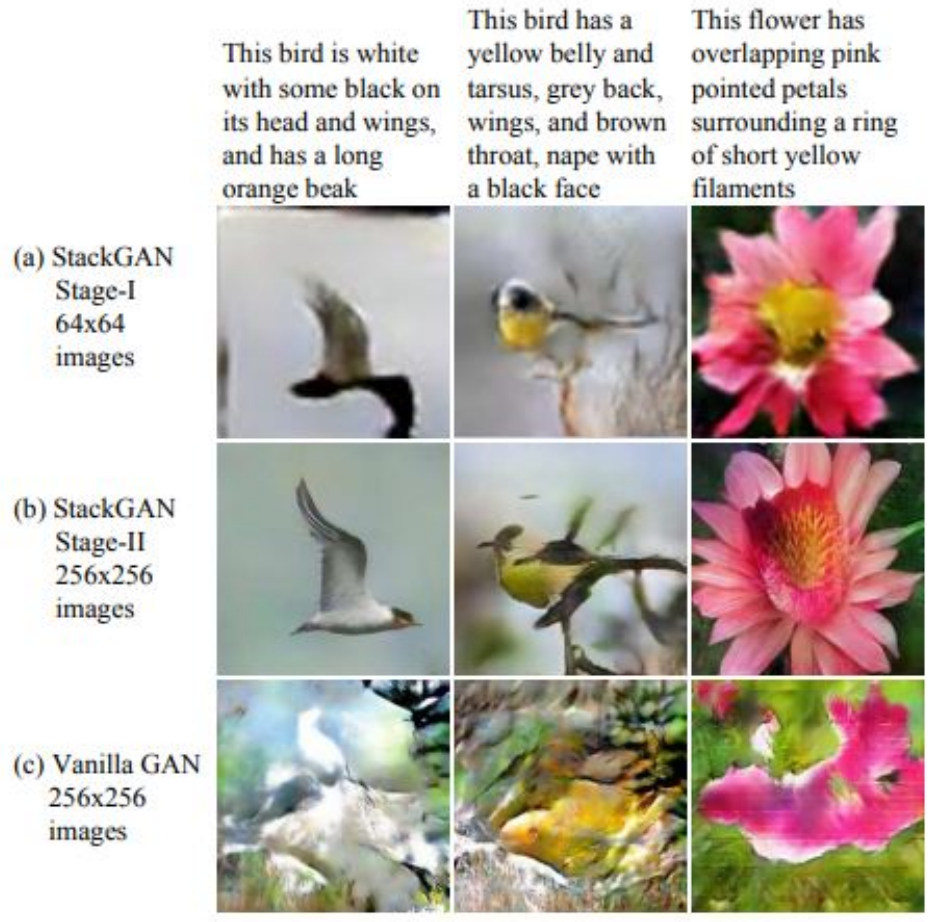


[Source: Accurate prediction of biological assays with high-throughput microscopy images and convolutional networks, Hofmarcher & Rumetshofer & Clevert & Hochreiter & Klambauer, JCIM 2019]

High-Resolution Image Classification

[Source: Classification of cell microscopy images, by courtesy of Rumetshofer & Kratzert]

Image Generation (GANs)



[Source: StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks, Zhang & Xu & Li & Zhang & Huang & Wang & Metaxas, ICCV 2017]

Summary

Summary

- State of the Art in all kinds of vision tasks
- Network is convolved over image
  - Image is divided into sub-images and processed by network with shared weights
- Spatial dimensionality can be reduced by striding or pooling
- Very large networks with many tweaks
- Dataset augmentation can be crucial