Deep Learning

Big Data & Machine Learning Bootcamp - Keep Coding

Outline

1. Classic networks
2. ResNets
3. Inception
4. Transposed Convolution
5. U-Net
6. GANs
7. Open-source implementations
8. Transfer learning
9. Data augmentation
10. State of computer vision

1. Classic networks

Similar to how we learn to write code, designing a new convolutional neural network can be learned by seeing what work for others.

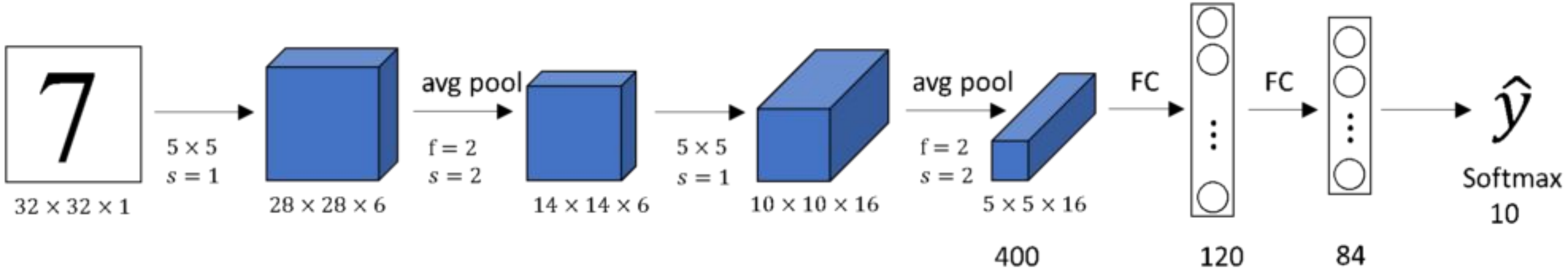
Examples of classic networks are:

- LeNet-5
- AlexNet
- VGG
- ResNet
- Inception

Sources:
- Coursera

1. Classic networks

LeNet-5 (Published in 1998)



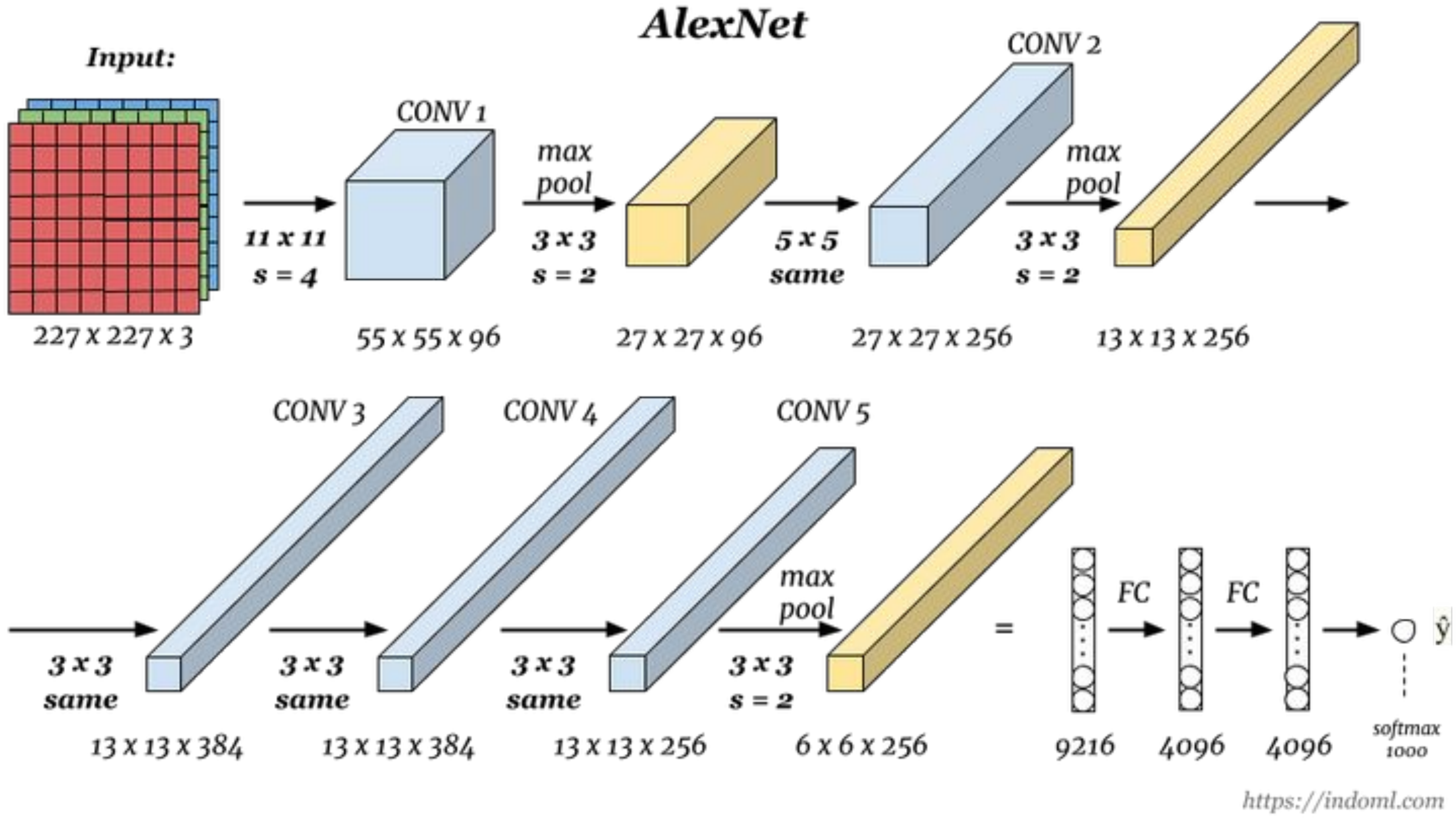
- This network is small by modern standards (Only 60.000 parameters. Today we often see networks with 10 Million to 100 Million of parameters
- Used a quite common pattern conv+pooling
- Used sigmoid and tanh activation functions instead of ReLU

LeCun, Yann, et al. "Gradient-based learning applied to document recognition." Proceedings of the IEEE 86.11 (**1998**): 2278-2324.

Sources:
- Coursera
- - http://datahacker.rs/deep-learning-lenet-5-architecture/

1. Classic networks

AlexNet (Published in 2012)



- Similar architecture to the LeNet-5 but much more bigger. ~60 Million parameters instead of 60.000
- Used ReLU activation function
- Trained in multiple GPUs

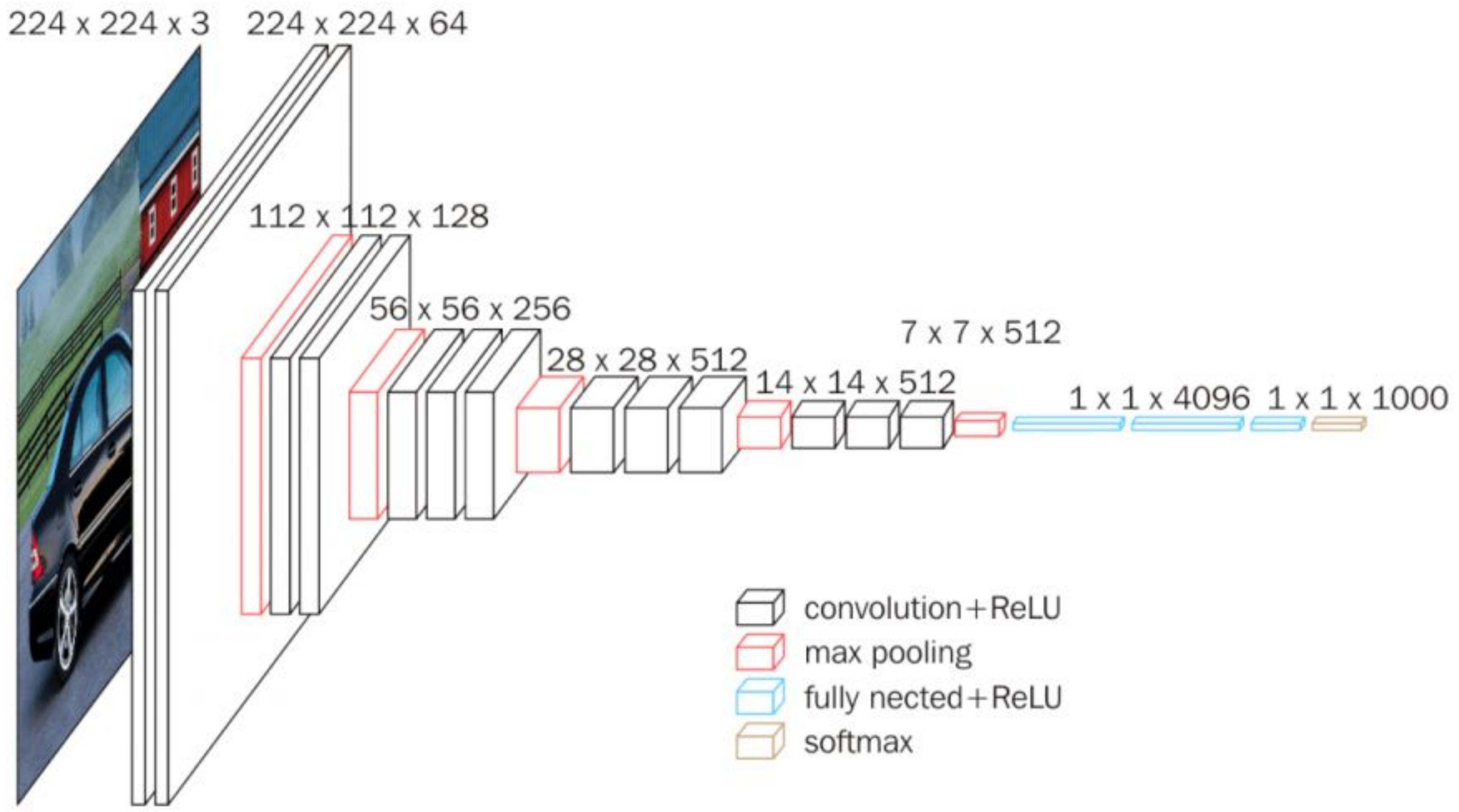
Thanks to the performance obtained from this network the computer vision community started to see CNNs more seriously!

Krizhevsky, Alex, Ilya Sutskever, and G. Hinton. "Imagenet classification with deep convolutional networks." Proceedings of the Conference Neural Information Processing Systems (NIPS). 2012

Sources:
- Coursera
- https://indoml.com/2018/03/07/student-notes-convolutional-neural-networks-cnn-introduction/

1. Classic networks

VGG16 (Published in 2014)



- The simplicity of this network is the its characteristic. conv+pool layers doubling the number of filters each layer
- Huge amount of parameters ~138 Million

Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

Sources:
- Coursera
- https://neurohive.io/en/popular-networks/vgg16/

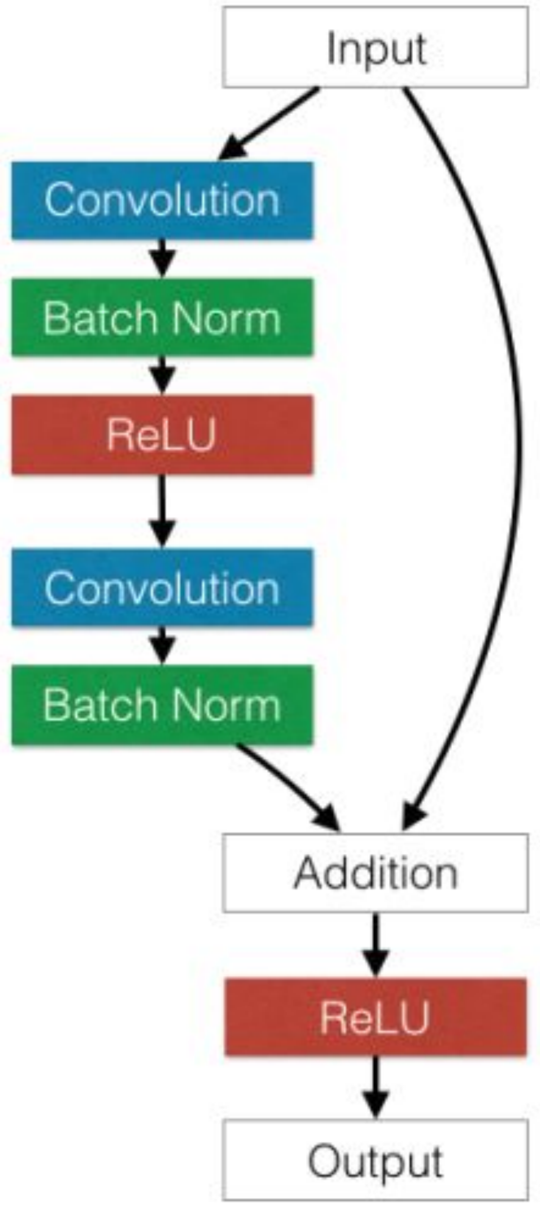
2. ResNets

Very very deep neural networks are difficult to train because of vanishing and exploding gradients.

The solution for that is to skip connections with something called the **residual block.**

It is just a way of organizing convolution layers!

The authors of this architecture found that using residual blocks allows them to train much more deeper networks (~152 layers)

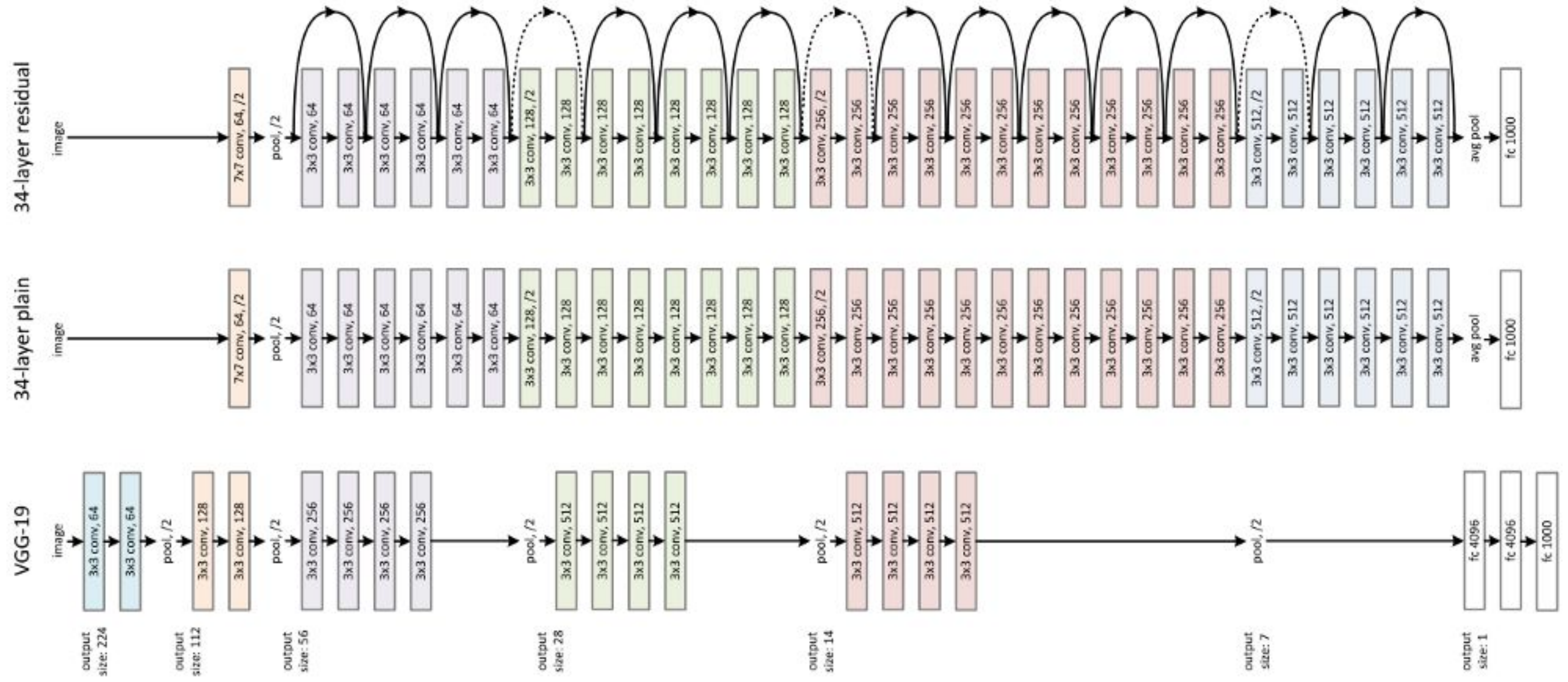


He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.

Sources:
- Coursera
- https://stackoverflow.com/questions/49293450/why-each-block-in-deep-residual-network-has-two-convolutional-layers-instead-of

2. ResNets

Plain network vs **Residual network**



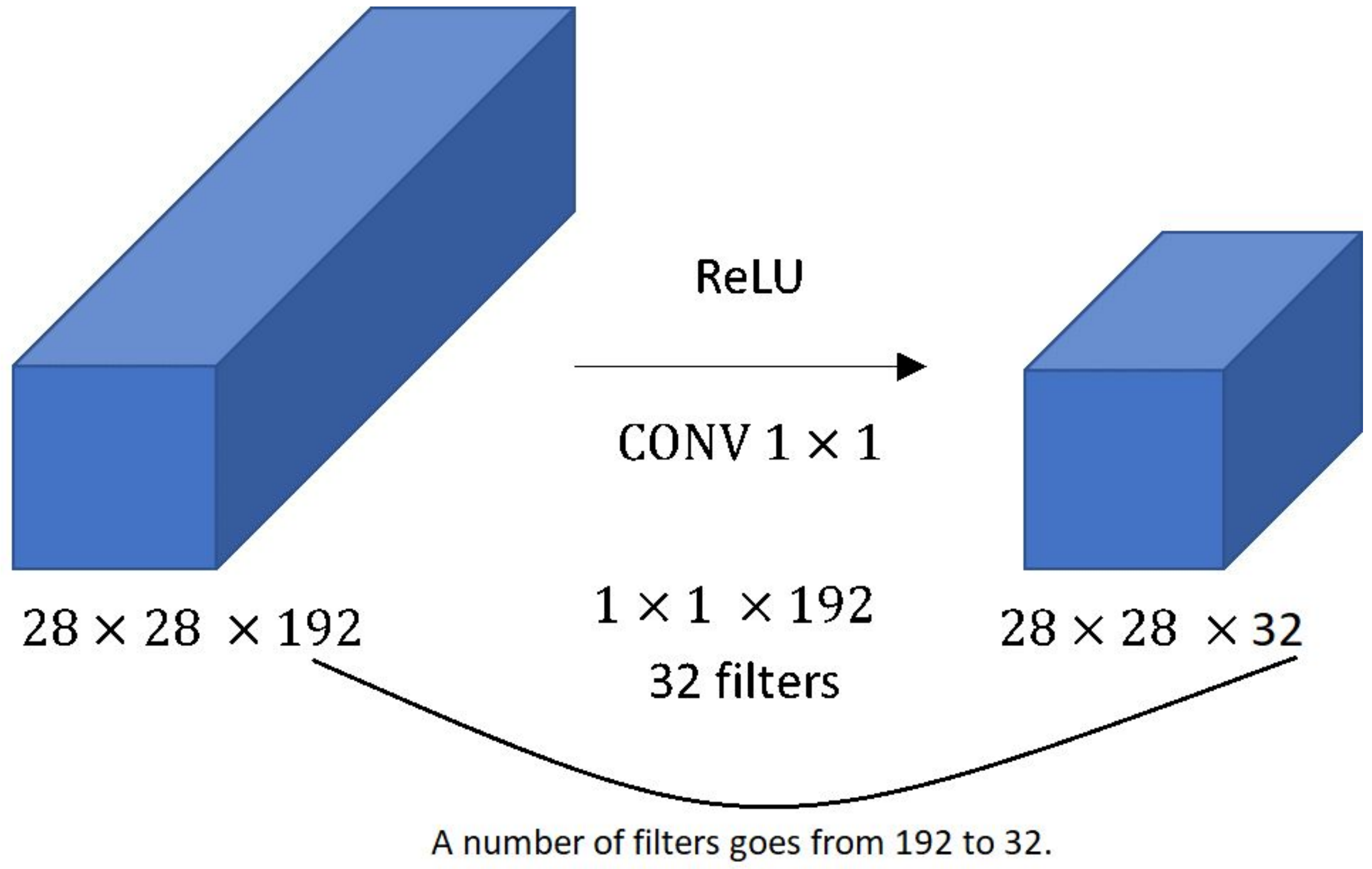
Sources:
- Coursera
- https://medium.com/analytics-vidhya/introduction-to-residual-neural-networks-8af5b7c4afd4

2. ResNets

Network-In-Network or 1x1 convolutions

In ResNets it is common to use **same convolutions** as the shape of the output should match with the next layer.

If the shapes don't match, there is something called **Network-In-Network or 1x1 convolutions** that shrink or increase the number of channels



Lin, Min, Qiang Chen, and Shuicheng Yan. "Network in network." arXiv preprint arXiv:1312.4400 (2013).

Sources:
- Coursera
- http://datahacker.rs/introduction-into-1x1-convolution/

3. Inception

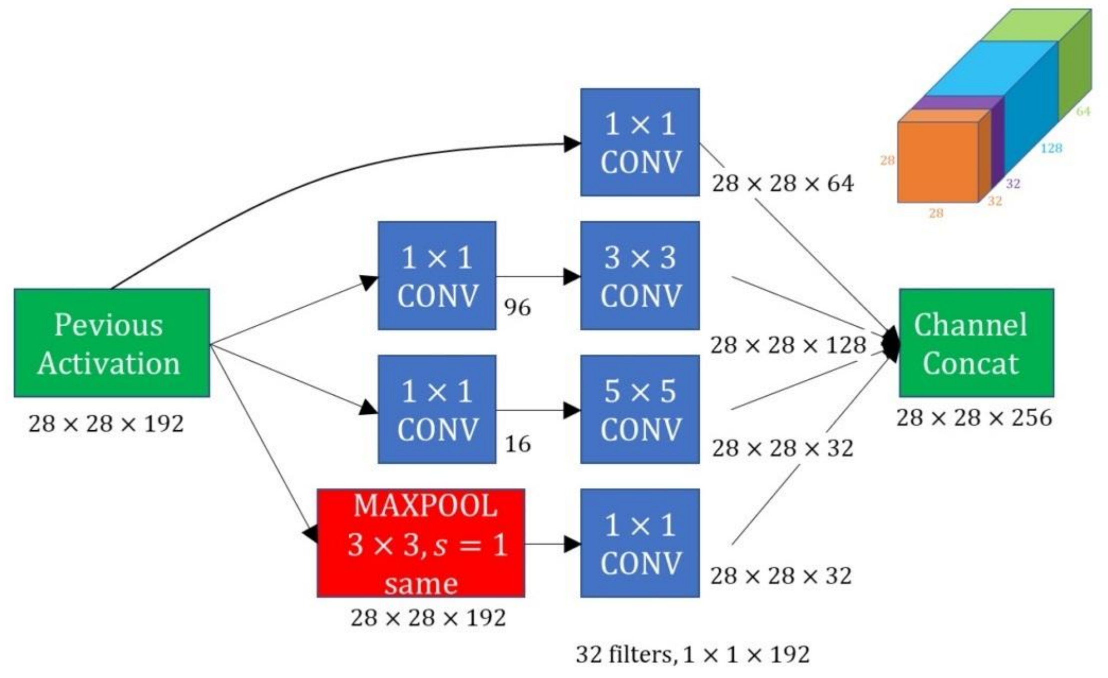
When designing a layer for a CNN, **we should pick if we want a layer with 3x3 filters**, 5x5 or 7x7.

BUT, what the Inception does is why not all of them?

That makes the network more complicated but also works very well!

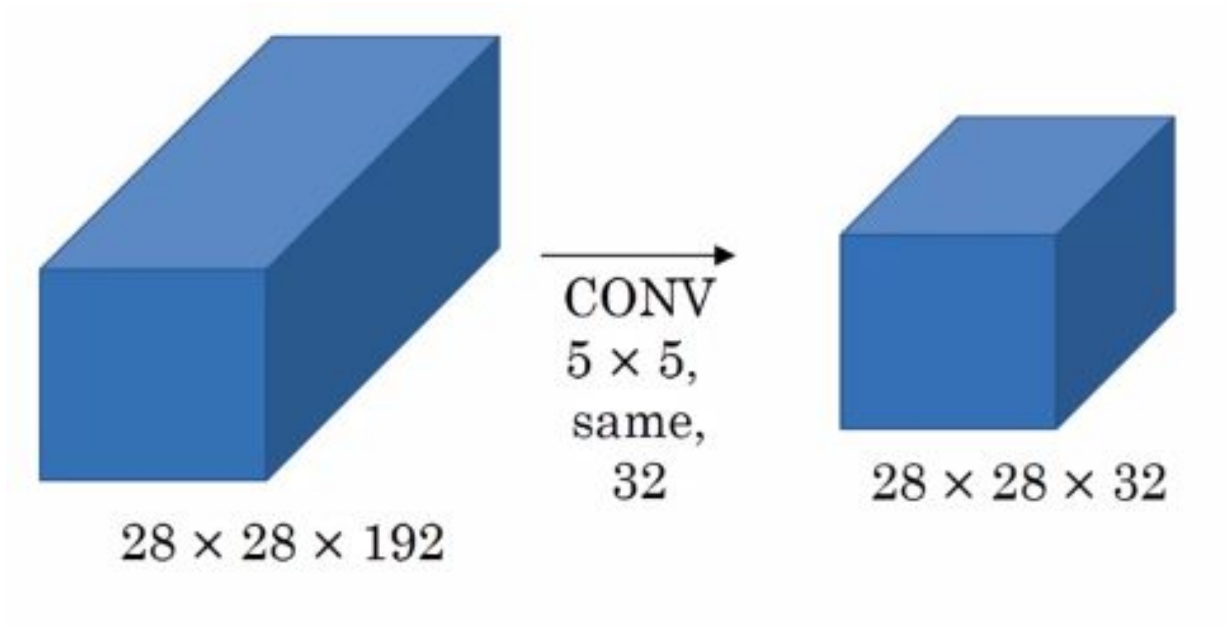
Don't choose! Let's use all of them!

An example of an Inception module



3. Inception

The problem may be the computation.



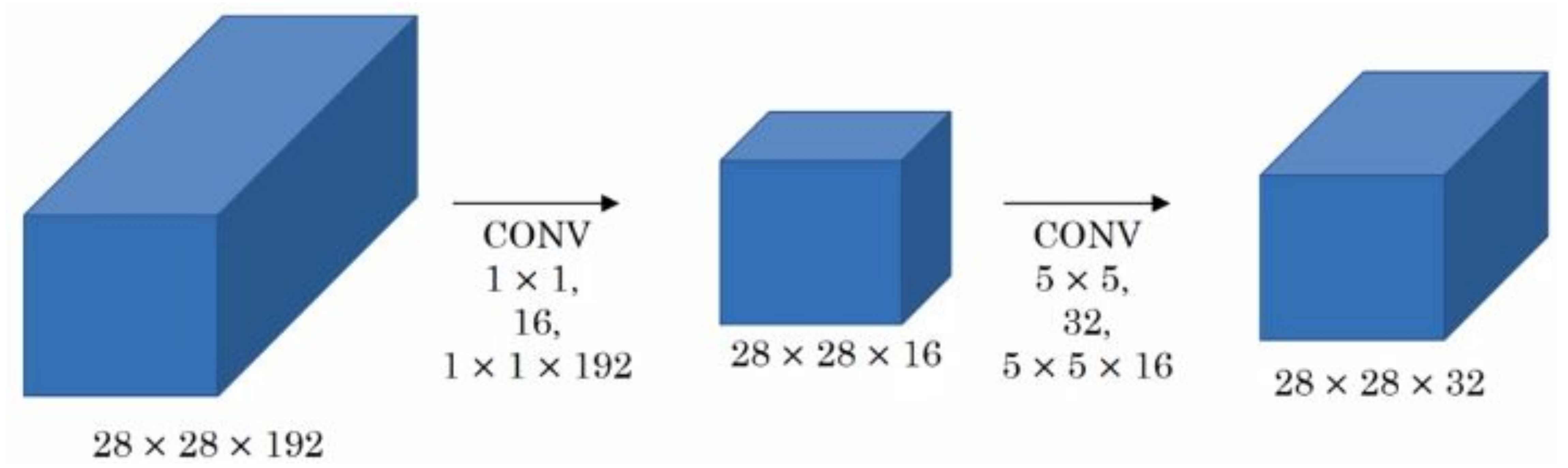
To output the volume on the right (28x28x32), we need to perform around 120 million multiplications.

Output volume x filter size
(28 x 28 x 32) x (5 x 5 x 192) ~ 120 million

Sources:
- Coursera

3. Inception

Reducing computation by using 1x1 convolution



From the same input 28x28x192, we can get the output volume of 28x28x32 with **around ~ 12.4 million multiplications**

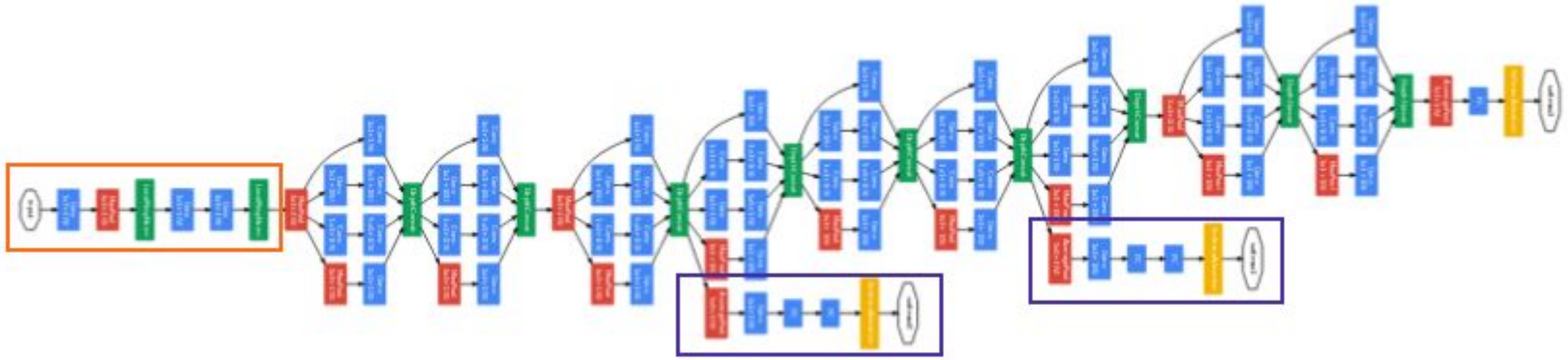
(28x28x16)x(1x1x192) = 2.4 million

(28x28x32) x (5x5x16) = 10 million

Sources:
- Coursera

3. Inception

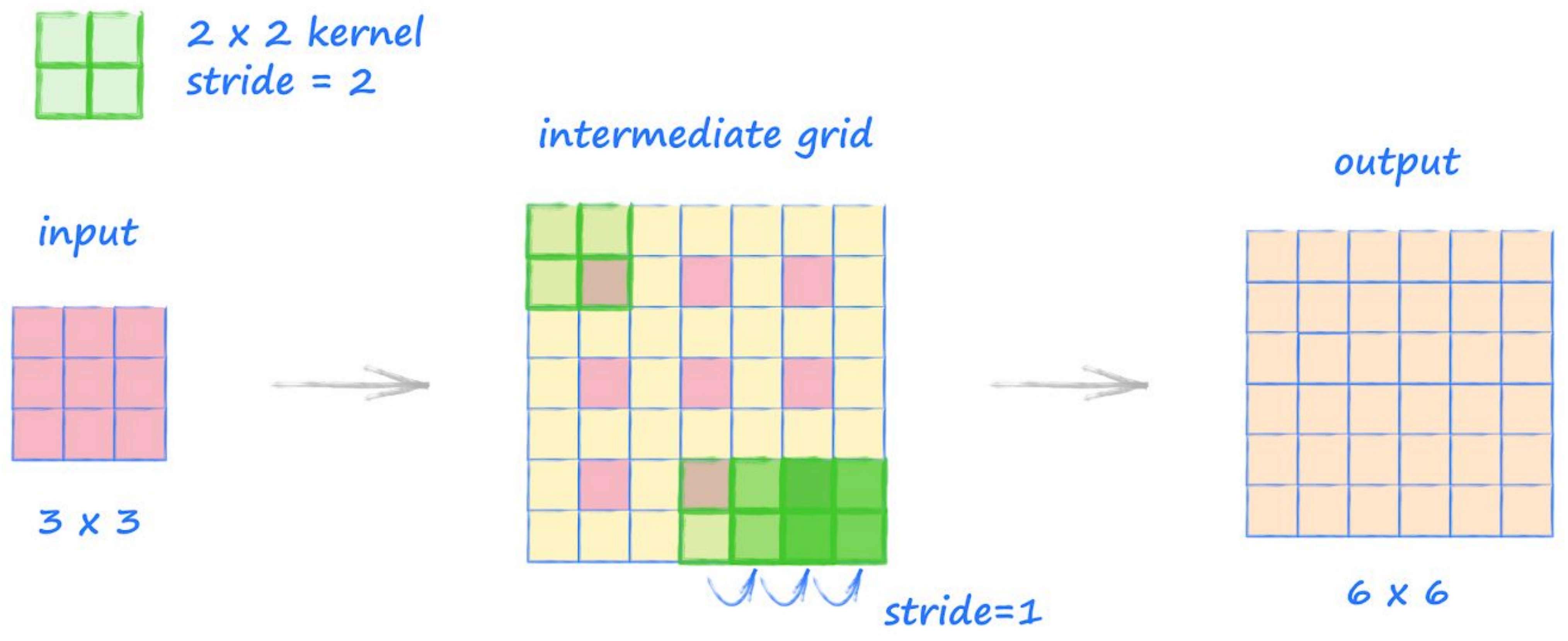
In case you were wondering, **that trick of using 1x1 conv doesn’t impact performance**.
Now the Inception module makes more sense, right? What about the Inception architecture?
Many Inception modules repeated!



Sources:
- Coursera
- https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202

4. Transposed Convolution

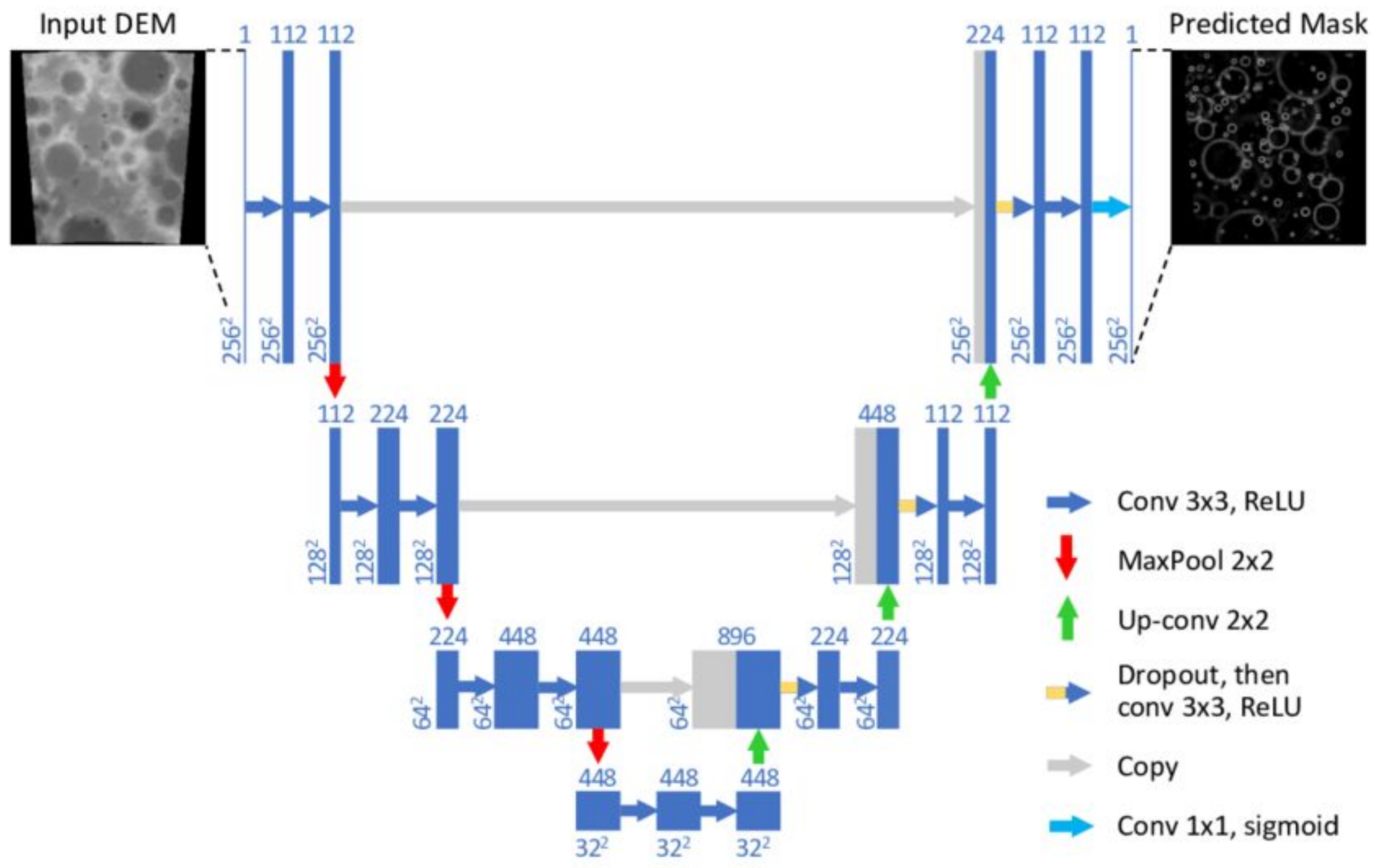
If we want our network to learn how to up-sample optimally, we can use the transposed convolution. It does not use a predefined interpolation method. **It has learnable parameters.**

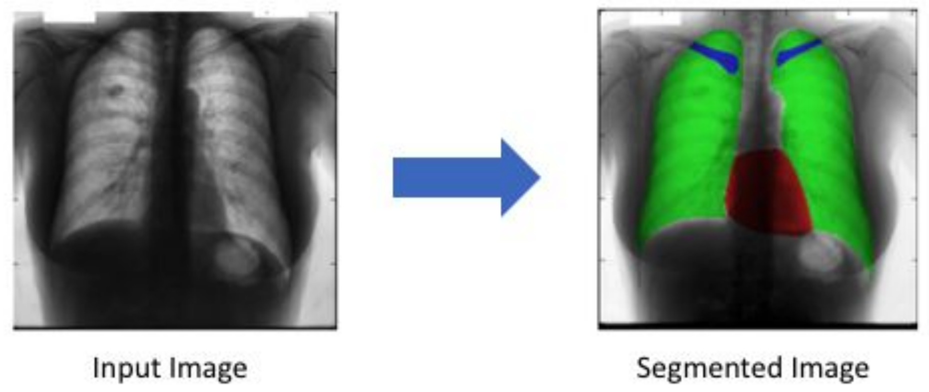


Sources:
- https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202
- http://makeyourownneuralnetwork.blogspot.com/2020/02/calculating-output-size-of-convolutions.html

5. U-Net - Segmentation Networks

UNet is a **convolutional neural network** architecture. It was designed to deal with biomedical images where the target is not only to classify whether there is an infection or not but also to identify the area of infection



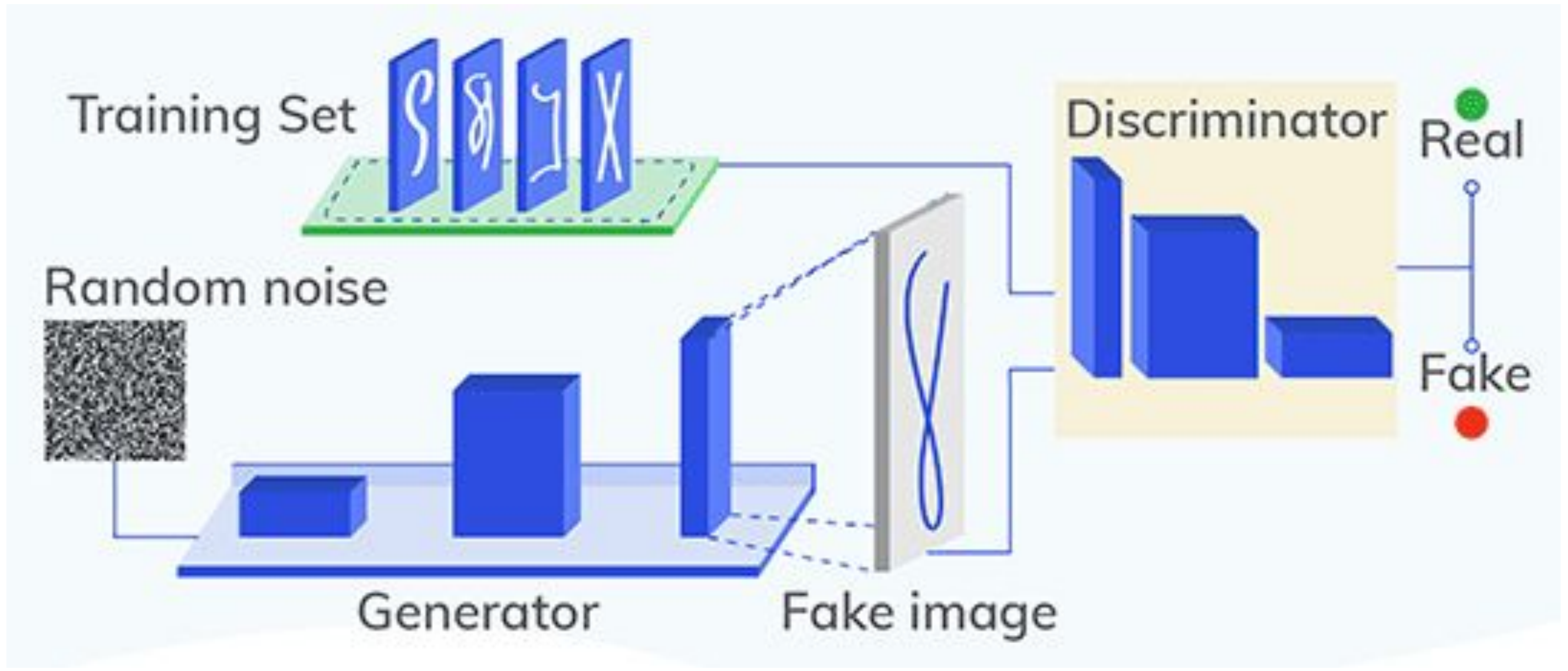


Sources:
- https://towardsdatascience.com/understanding-semantic-segmentation-with-unet-6be4f42d4b47
- https://arxiv.org/abs/1803.02192

6. Generative Adversarial Networks (GANs)

GANs is a generative model that uses deep learning methods, such as convolutional neural networks.

Generative modeling is an **unsupervised learning task** in machine learning that involves automatically discovering and learning the regularities or patterns in input data in such a way that the model can be used to generate or output new examples that plausibly could have been drawn from the original dataset.



Sources:
- https://pubmed.ncbi.nlm.nih.gov/30843823/
- https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/
- https://github.com/soumith/ganhacks

7. Open-source implementations

Practical advice: Use the open-source implementations

Implementing CNNs from reading the article is really challenging, even for researchers. Fortunately there is people that implemented and made them available to us.

VGG: TensorFlow: https://github.com/tensorflow/models/blob/master/research/slim/nets/vgg.py
PyTorch: https://github.com/pytorch/vision/blob/master/torchvision/models/vgg.py

ResNet: TensorFlow: https://github.com/ry/tensorflow-resnet
PyTorch: https://github.com/pytorch/vision/blob/master/torchvision/models/resnet.py

Inception: TensorFlow: https://github.com/tensorflow/models/blob/master/research/slim/nets/inception_v3.py
PyTorch: https://github.com/pytorch/vision/blob/master/torchvision/models/inception.py

Sources:
- Coursera

8. Transfer learning

Instead of training CNN from scratch using random initialization, we can make more progress by **using the weights that someone else obtained from training a neural network.**

In the computer vision community there are **many publicly available datasets** such as ImageNet, COCO, PASCAL, etc.

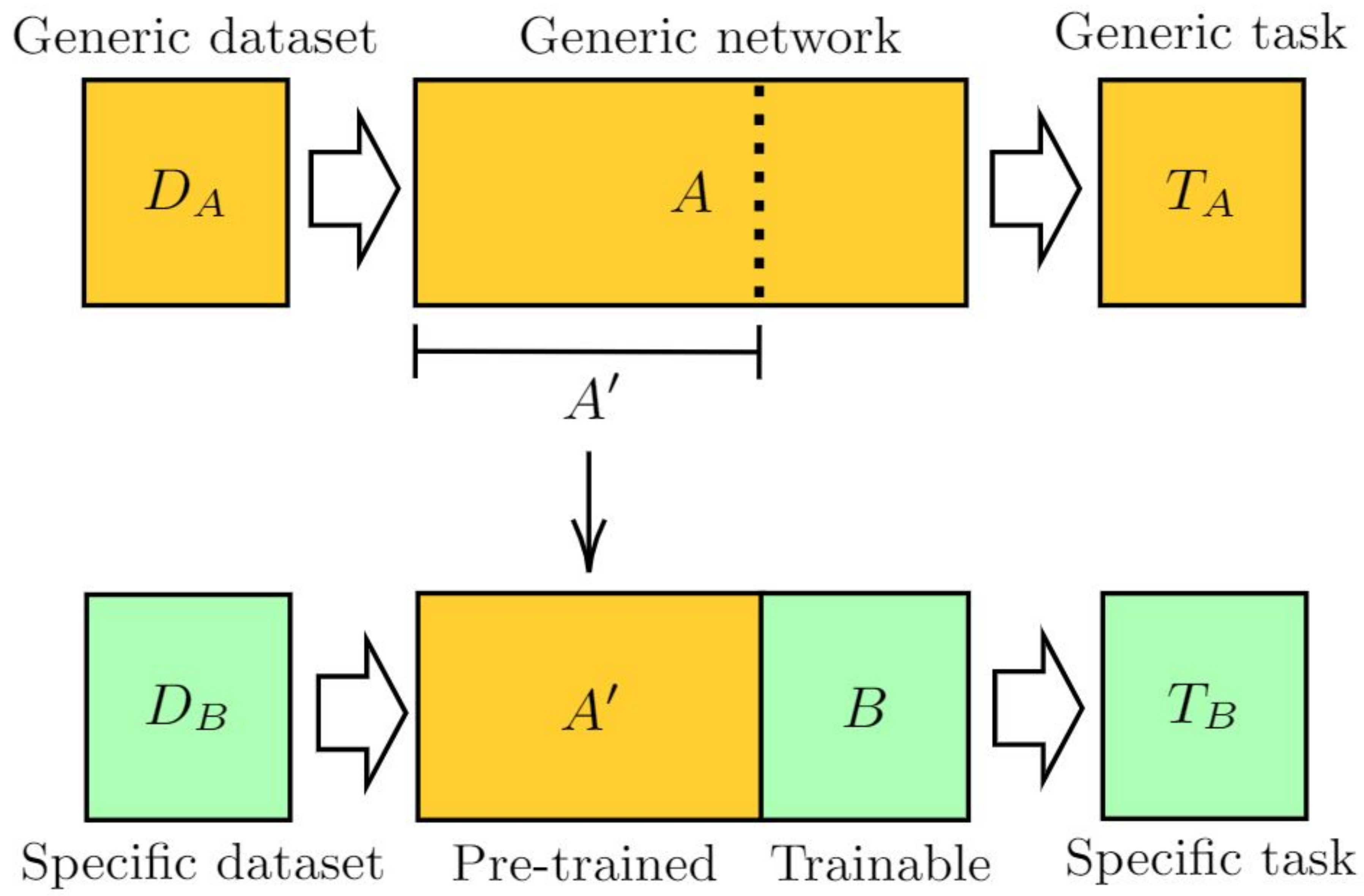
The availability of these networks helped a lot to advance on computer vision as they **allow other researchers to use pre-trained CNNs on different tasks**

An example is a study published during my PhD on predicting Glaucoma using retinal images: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6425593/

Sources:
- Coursera
- https://en.wikipedia.org/wiki/Fundus_photography

8. Transfer learning

I'd recommend downloading the architecture and its weights



Change the softmax layer for the number of classes you want to classify.

You can freeze some layers and make others trainable.

How many layers you should freeze?
It depends on how big is the dataset you have for the specific task. **The more images you have the more layers you can retrain.**

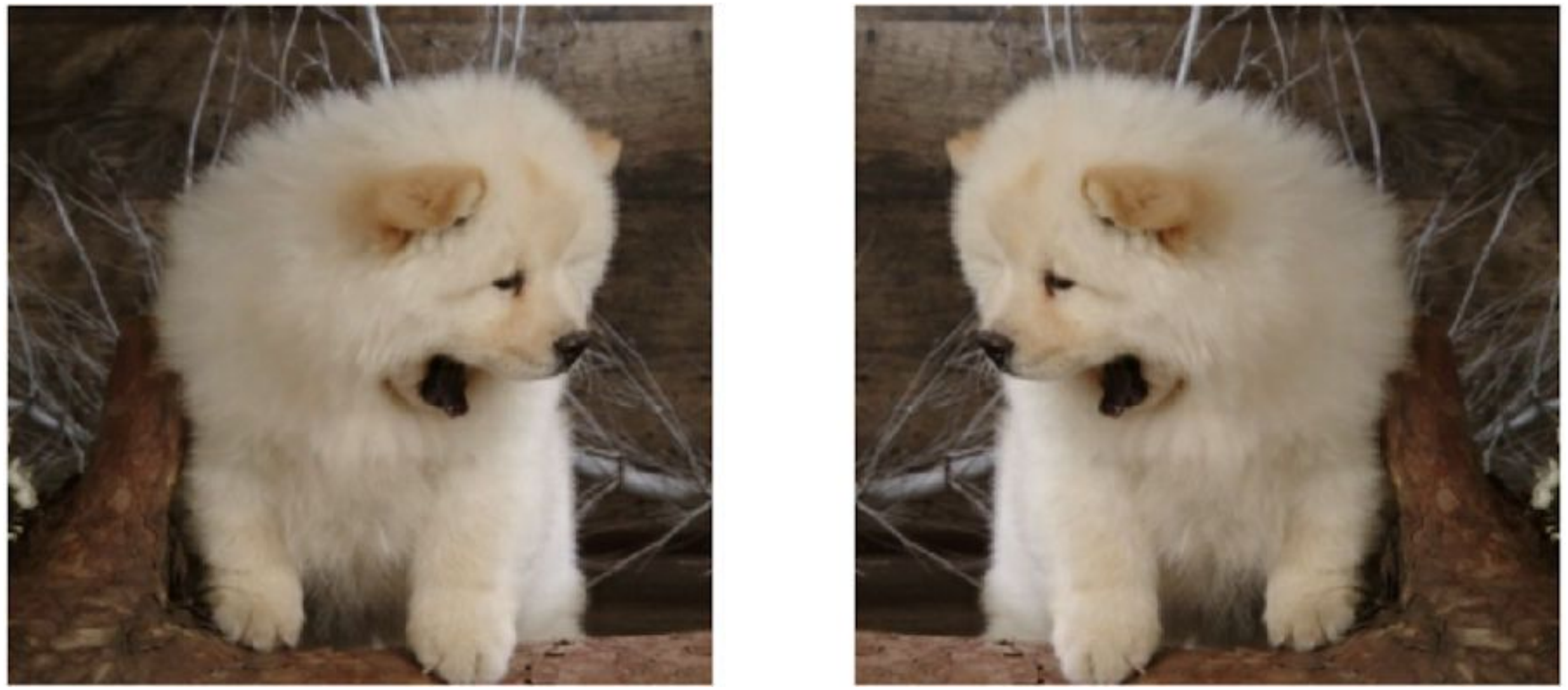
Most of the deep learning frameworks allow you to do this fairly easy.

Sources:
- Coursera
- https://pennylane.ai/qml/demos/tutorial_quantum_transfer_learning.html

9. Data augmentation

It is one of the most useful techniques to increase the number of images to train/retrain computer vision models. It is applied to all tasks in which labels are not modified after transforming the images

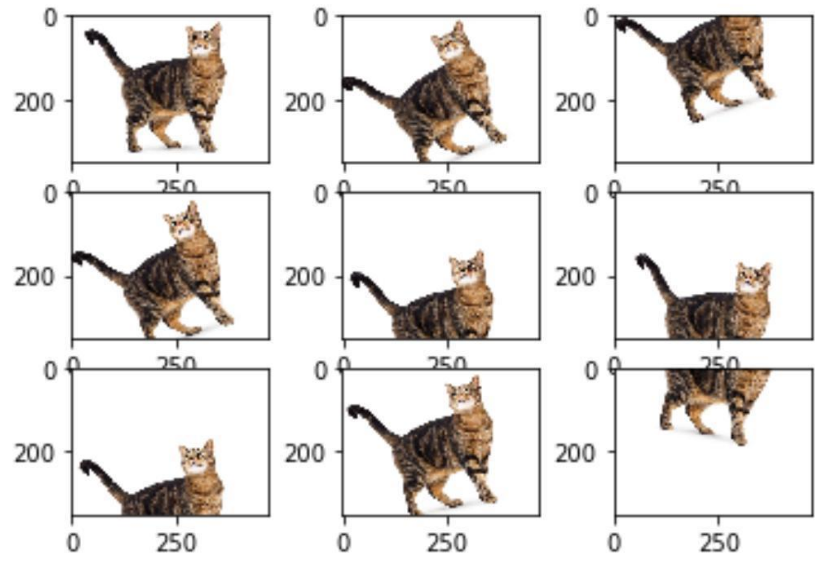
- Mirroring
- Random cropping
- Rotation and/or shifting
- Color shifting



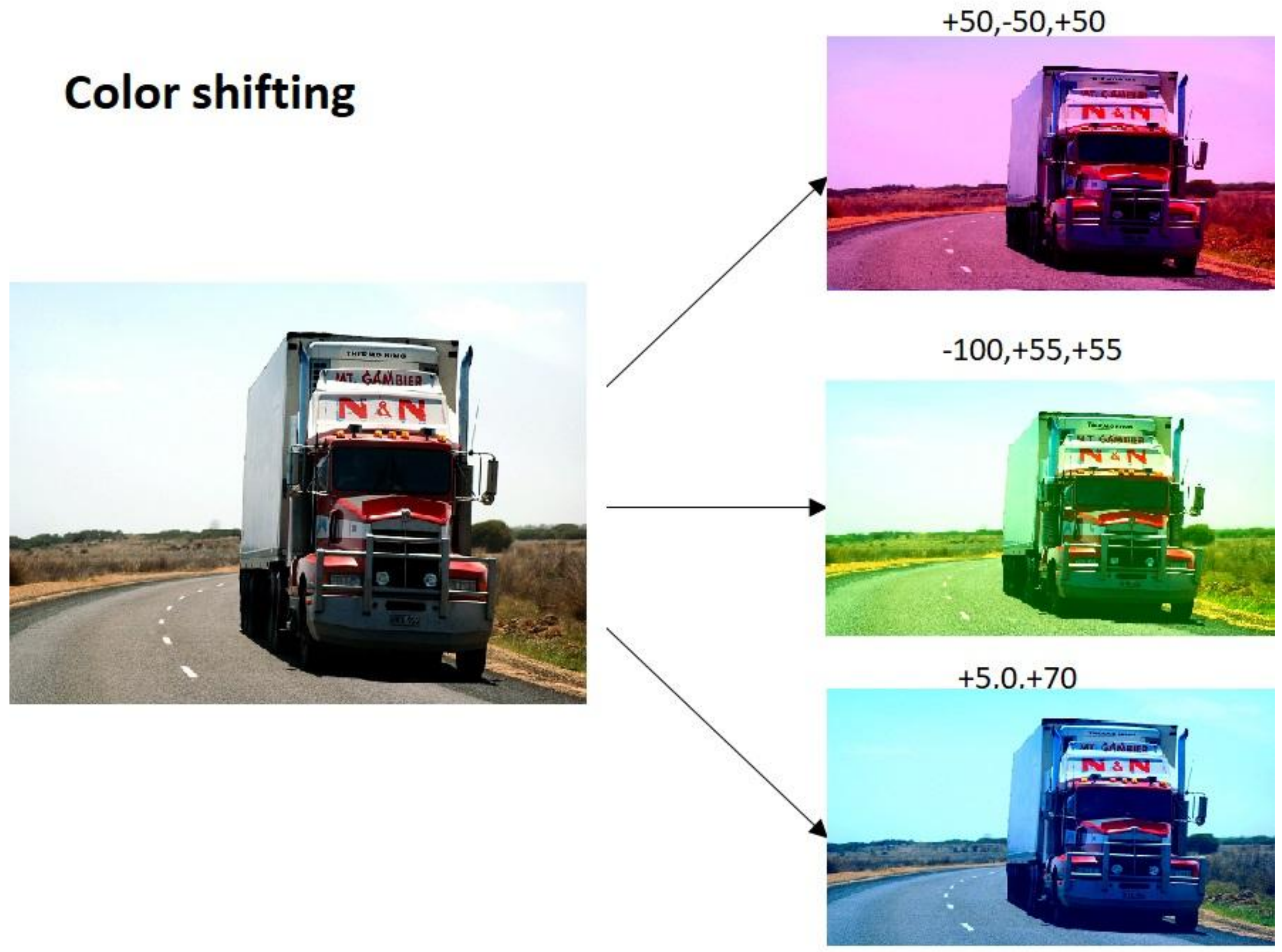
After mirroring, it is the same dog, isn't it?

Sources:
- Coursera
- https://medium.com/@rahuladream/multiply-your-dataset-using-data-augmentation-techniques-381aee8ff8f6

9. Data augmentation



Rotation and/or shifting



Color shifting

Sources:
- Coursera
- http://datahacker.rs/deep-learning-data-augmentation/

9. Data augmentation

A good thing to do is to perform the **data augmentation "online"**

While loading the original images, and before passing them to the model, data modification is performed in parallel

Don't need to save them in disc

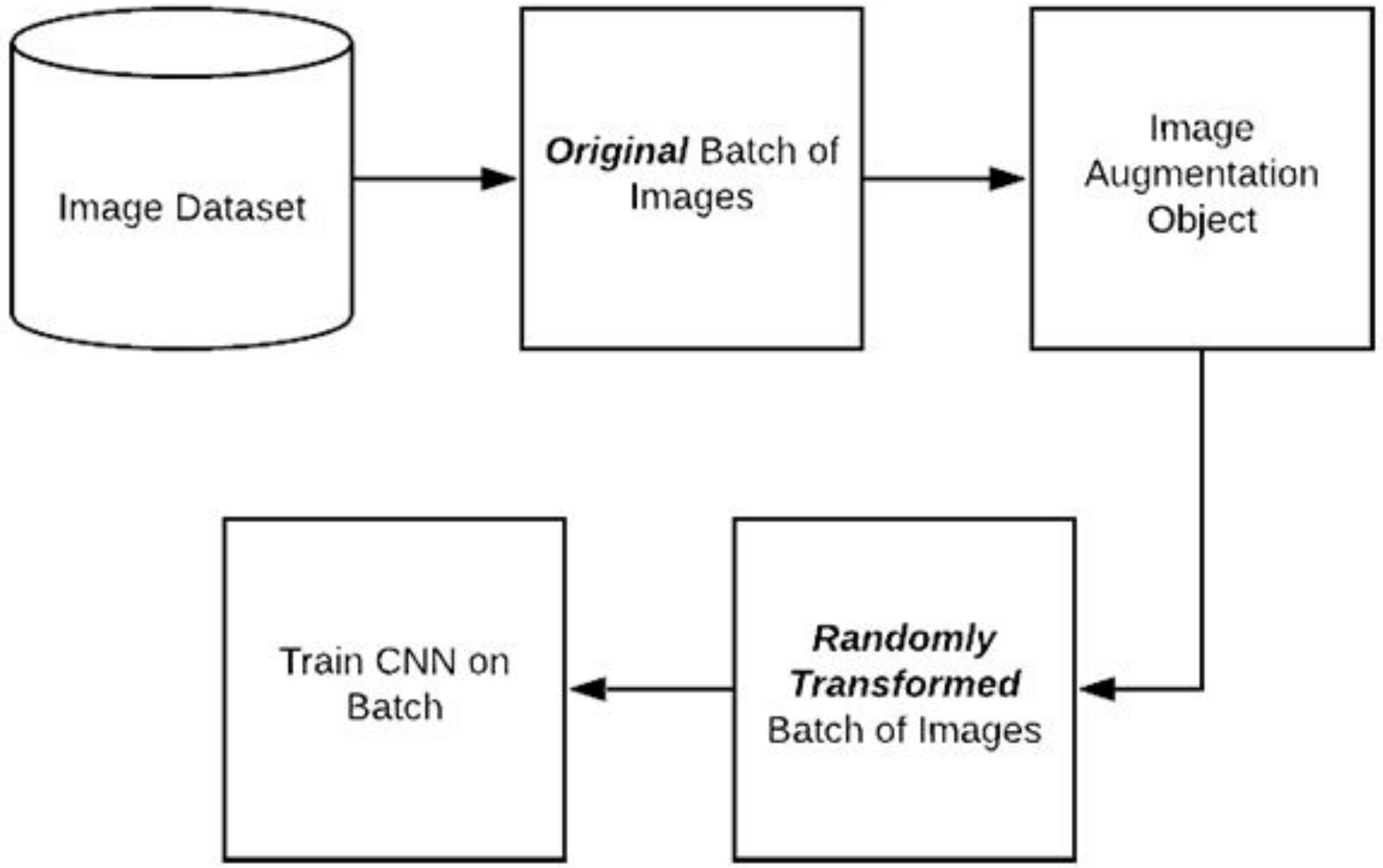


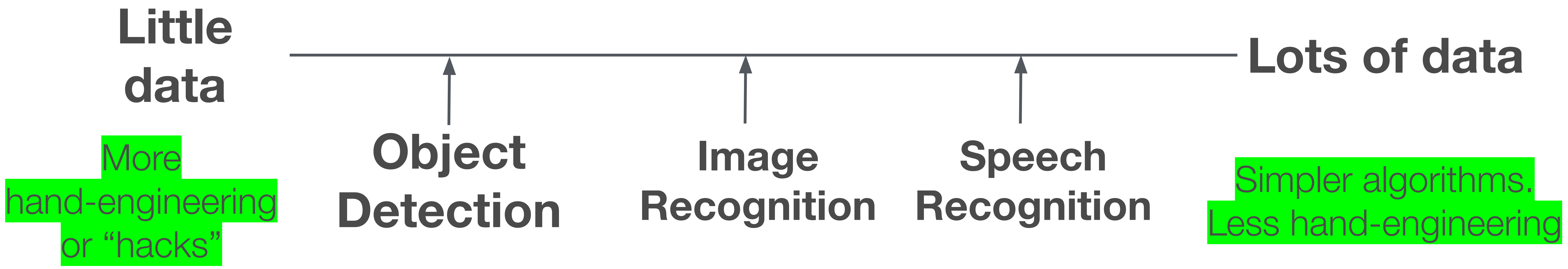
Image Dataset
Original Batch of Images
Image Augmentation Object
Randomly Transformed Batch of Images
Train CNN on Batch

Sources:
- Coursera
- https://www.pyimagesearch.com/2019/07/08/keras-imagedatagenerator-and-data-augmentation/

10. State of computer vision

Most of machine learning problems fall on the spectrum where you have little data and lots of data.

For instance, today there is already lots of data for speech recognition



There are two sources of knowledge:

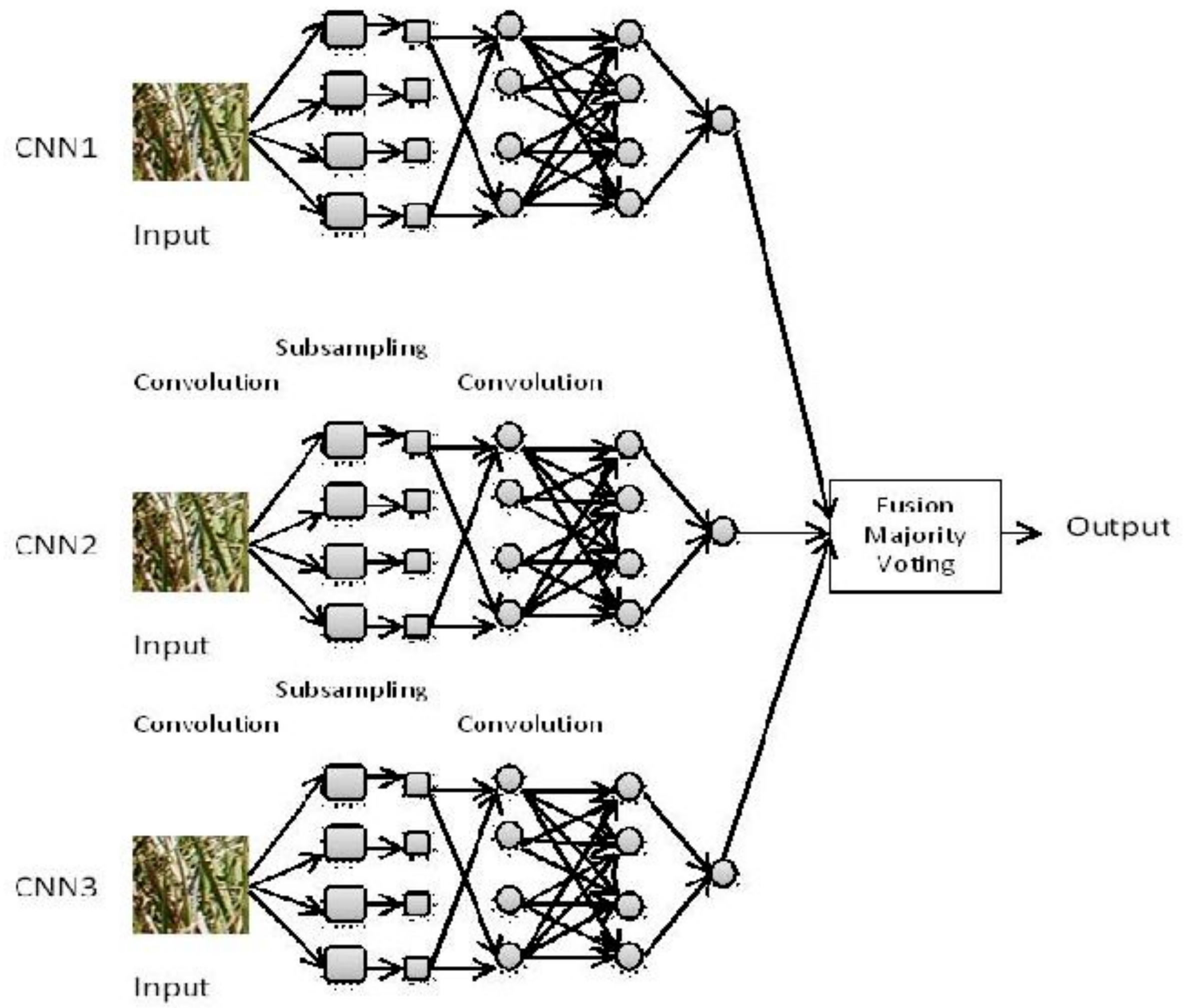
- Labeled data
- Hand engineered features/network architecture/other componentes

Sources:
- Coursera

10. State of computer vision

Tips to win competitions:

- **Ensembling:** Train several networks independently and average their outputs
- **Multi-crop at test time:** Run classifier on multiple versions of test images and average results



Sources:
- Coursera
- https://arxiv.org/pdf/2002.09103.pdf

Let’s move to Google Colab!

Notebooks:

- *7_CNN_Emotion_Detection.ipynb*
- *GAN_tutorial.ipynb*

Sources:
- Coursera