Real-world use cases of Deep Learning for Image Processing

Vitalii Duk

Vitalii Duk

- **Currently:** *Lead Data Scientist* @ EMLC.ai
- **Previously**: *Senior Data Scientist* @ dubizzle
- Co-organizer of **Dubai Data Science meetups**
- https://www.linkedin.com/in/vit-d
- vitalii.duk@emlc.ai

2nd annual Dubai Data Science Meetup @ AREA 2071

http://dubaidatascience.ae

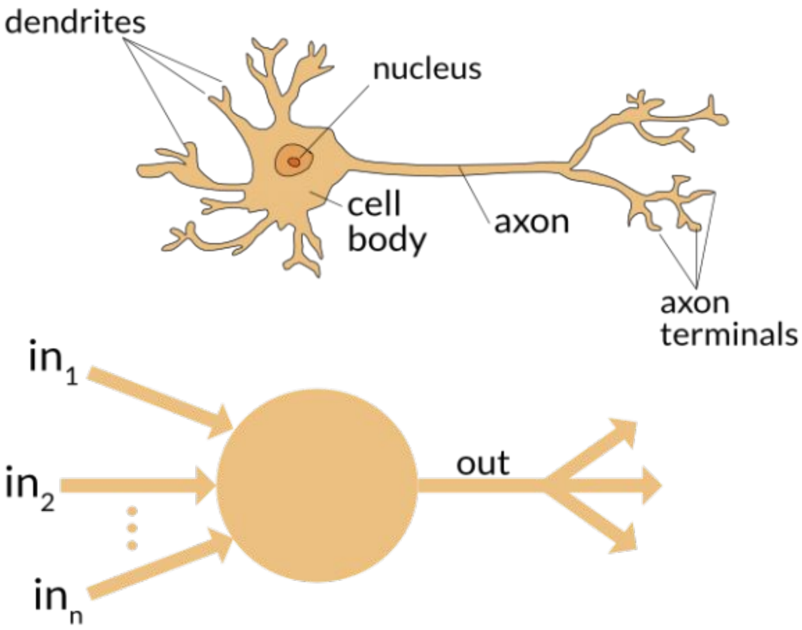
About ODS Dubai:

- the biggest Data Science community in UAE
- ~ **1200** group members
- ~ **90** participants every event
- completely free of charge

Basics of
Artificial Neural Networks

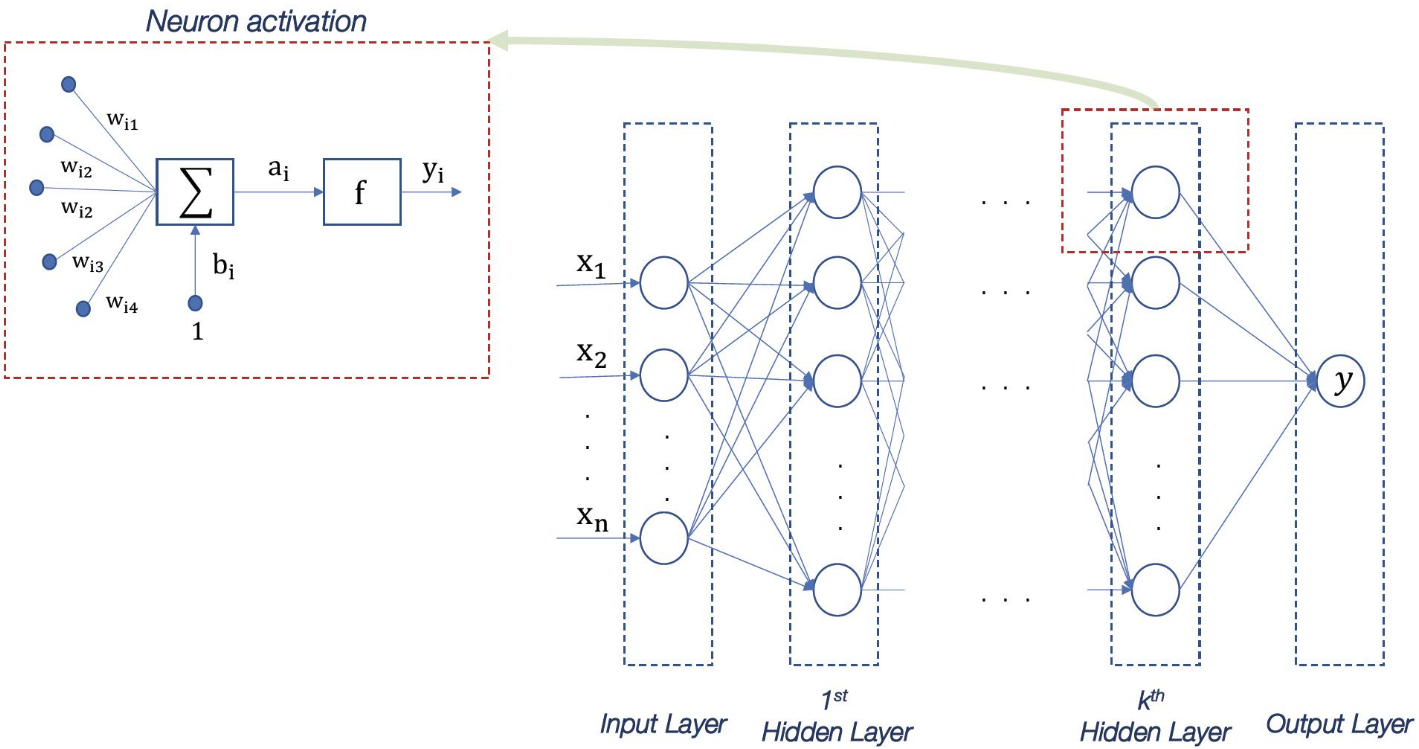
Artificial Neural Networks

- Computing systems vaguely inspired by the biological neural networks that constitute animal brains.

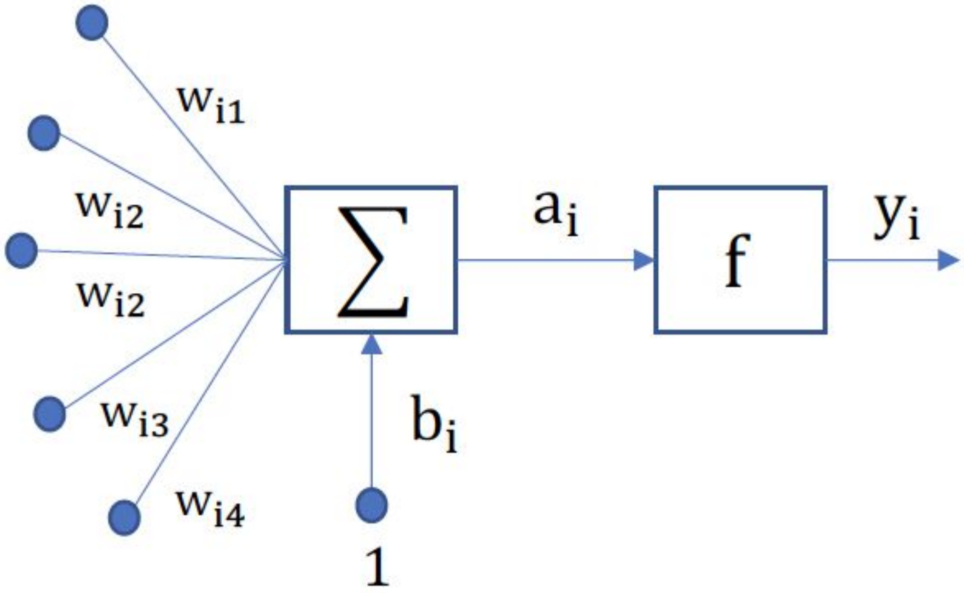


Artificial Neural Networks

- The multilayer perceptron is a universal function approximator, as proven by the universal approximation theorem.

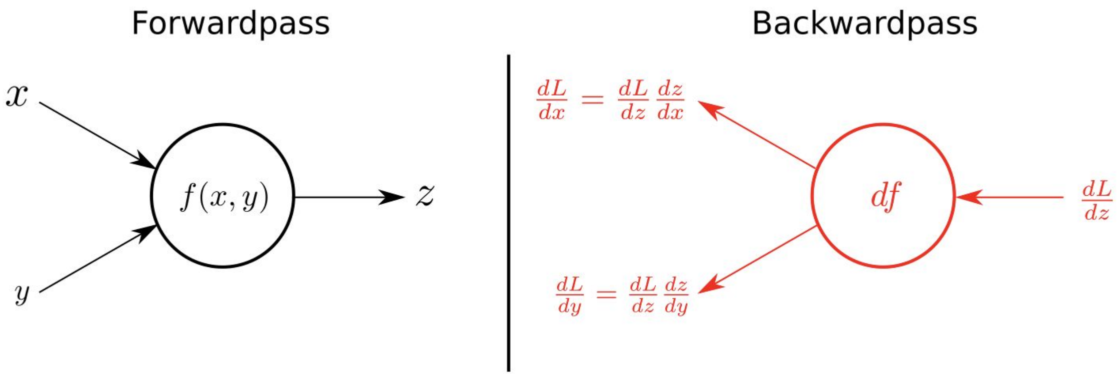


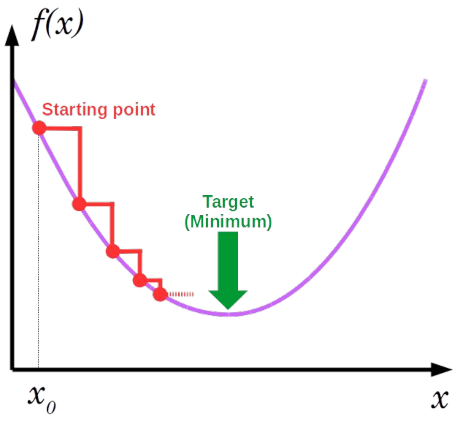
Neuron looks like … Logistic Regression



$$p = \frac{1}{1 + e^{-(b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_p x_p)}}$$

Artificial Neural Networks - backpropagation (1960's - 1980's)





Building blocks of Convolutional Neural Networks

How does the computer see an image?

- RGB representation

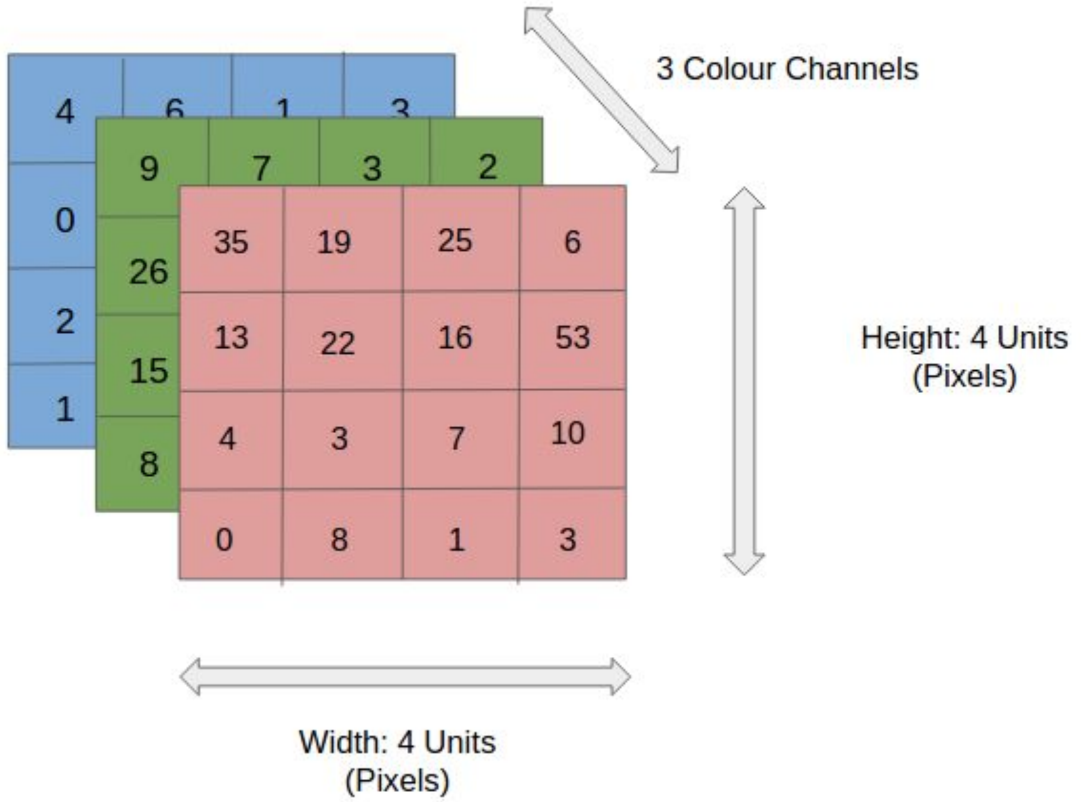
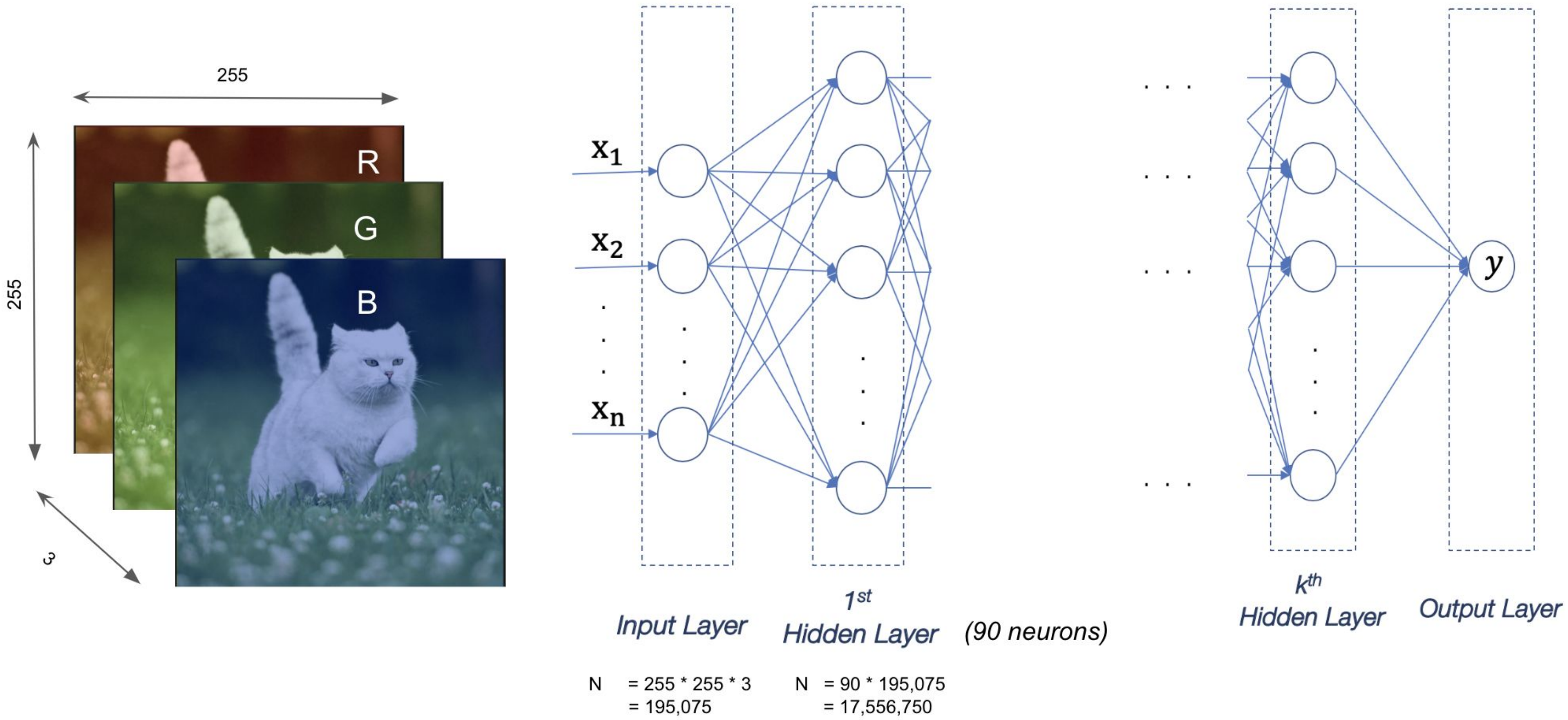
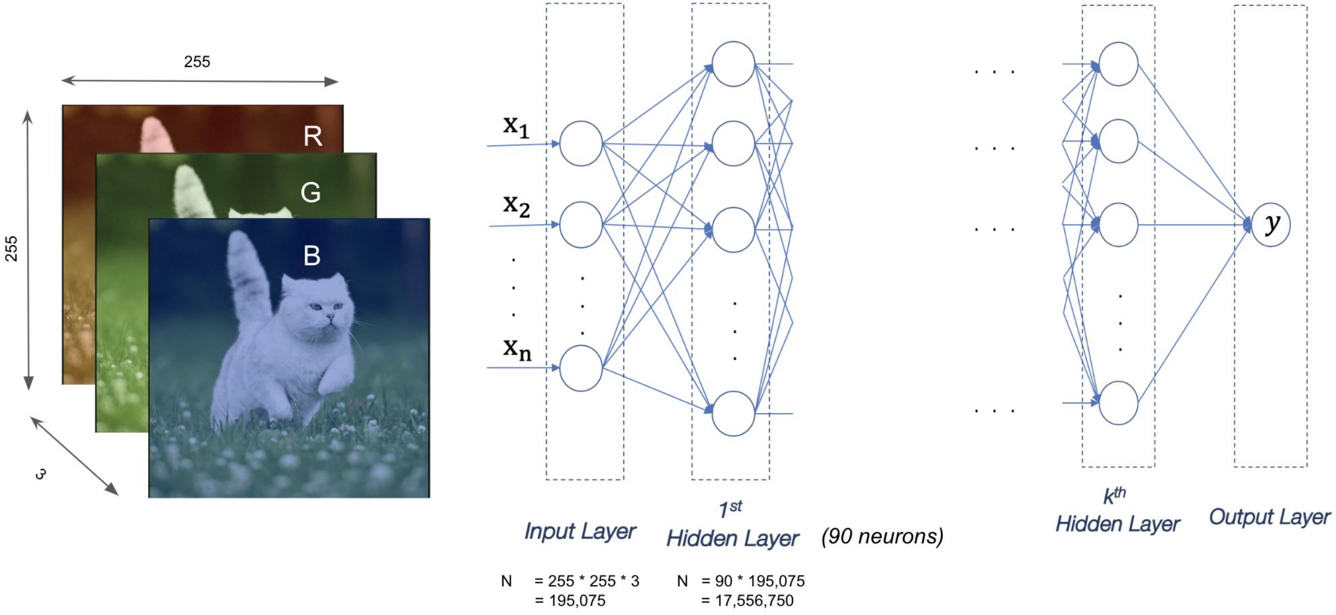


Image classifier using Fully-Connected Neural Network



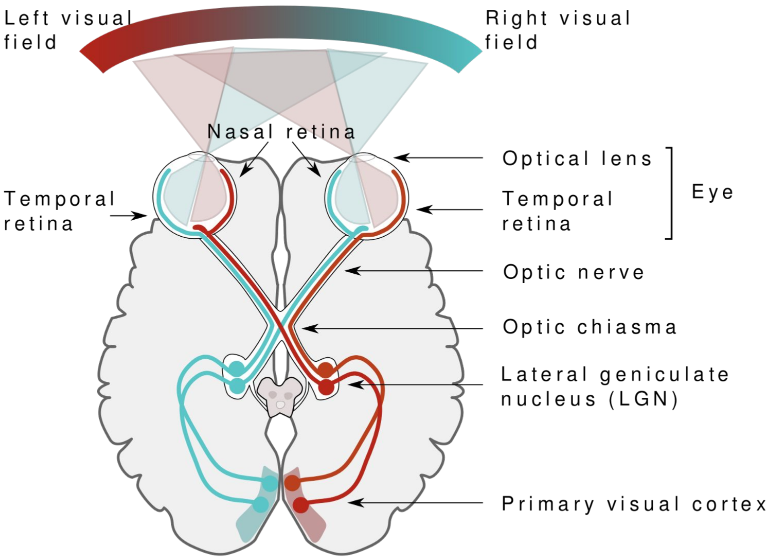
Main issues with FC networks for image classification

- **a huge** number of parameters to learn (more than 17M in the example below)
- **not invariant** to small transformations or distortions of the input image
- tend to **overfit**



Convolutional Neural Networks

- were inspired by **biological processes** in that the connectivity pattern between neurons resembles the organization of the animal **visual cortex**.
- individual cortical neurons respond to stimuli only in a **restricted region** of the visual field known as the **receptive field**
- the receptive fields of different neurons partially **overlap** such that they cover the entire visual field.



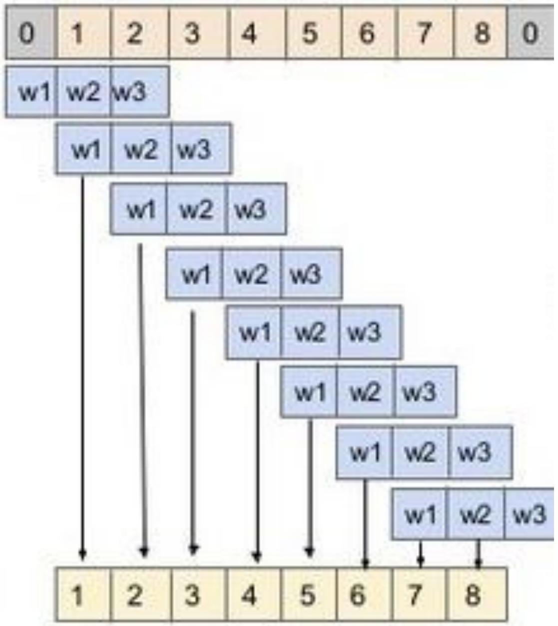
Convolutional Neural Networks

- Work by Hubel and Wiesel in the 1950s and 1960s showed that cat and monkey visual cortexes contain neurons that individually respond to small regions of the visual field.
- Kunihiko Fukushima in the 1980s proposed the **neocognitron** - a hierarchical, multi-layered artificial neural network used for handwritten character recognition. The neocognitron was inspired by the model proposed by Hubel & Wiesel.
- LeCun et in 1989, used back-propagation to learn the **convolution kernel** coefficients directly from images of hand-written numbers.
- In 1998 LeCun developed LeNet-5 - a pioneering 7-level convolutional network.
- 2012 was the first year that Convolutional Neural Networks grew to prominence as **Alex Krizhevsky** used them to win that year's **ImageNet** competition.

Convolutions

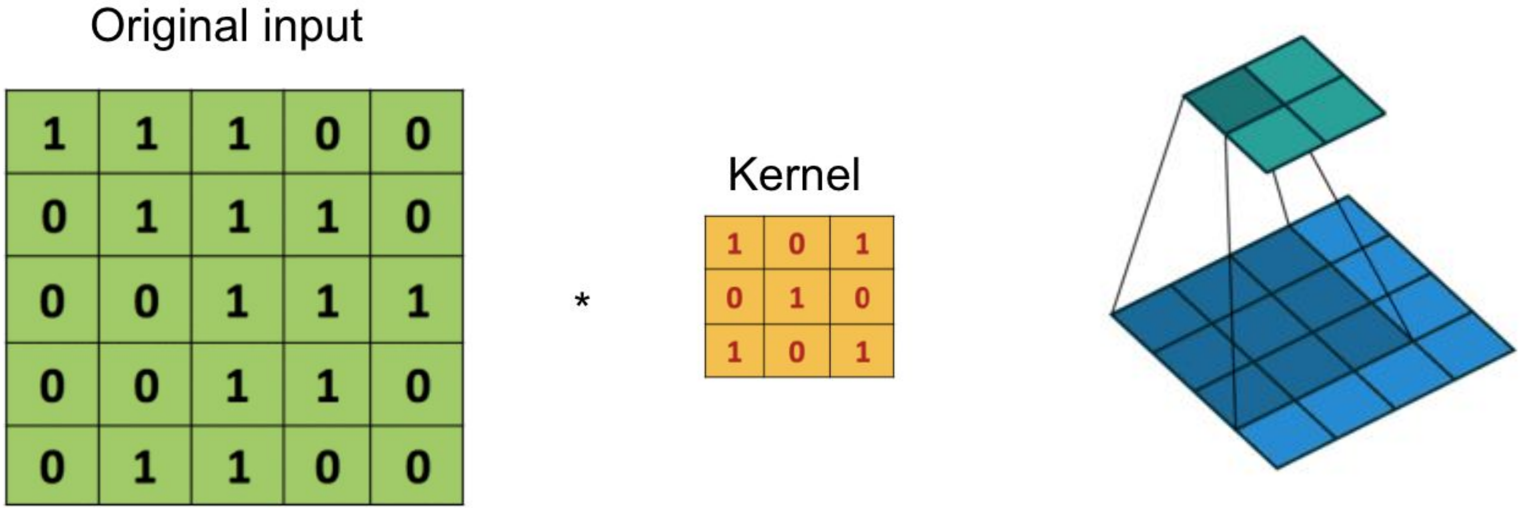
- **convolution operation** is an element-wise product and sum between two matrices/vectors - **kernel** or **filter** and part of the **original input** or **image**.

1D convolution



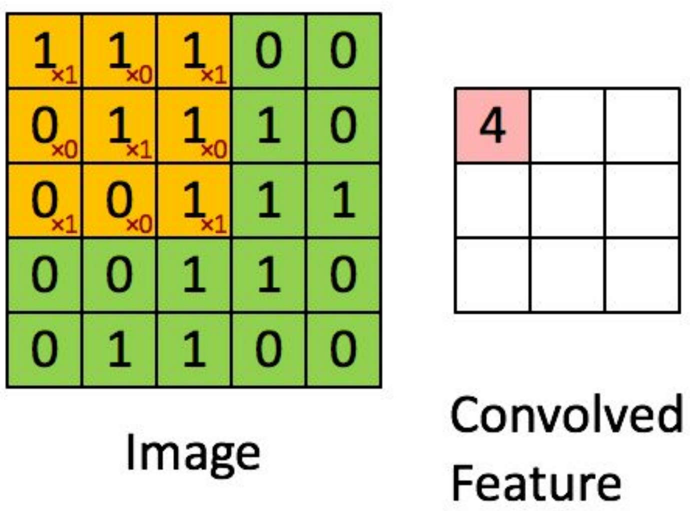
Convolutions

2D convolution



Convolutions

- The **convolution** of the **5x5 image** and the **3x3 kernel** can be computed as shown in the animation.
- We slide the **kernel** (orange matrix) over our **original image** (green) by 1 pixel (also called **stride**) and for every position and for every position, we compute element-wise multiplication (between the two matrices) and add the multiplication outputs.
- The **3×3 kernel** observes only a part of the input image in each stride.
- **Kernel** acts as a **feature detector** from the original input image.



Convolutions - more examples

Edge detection

$$* \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \quad =$$

Gaussian blur

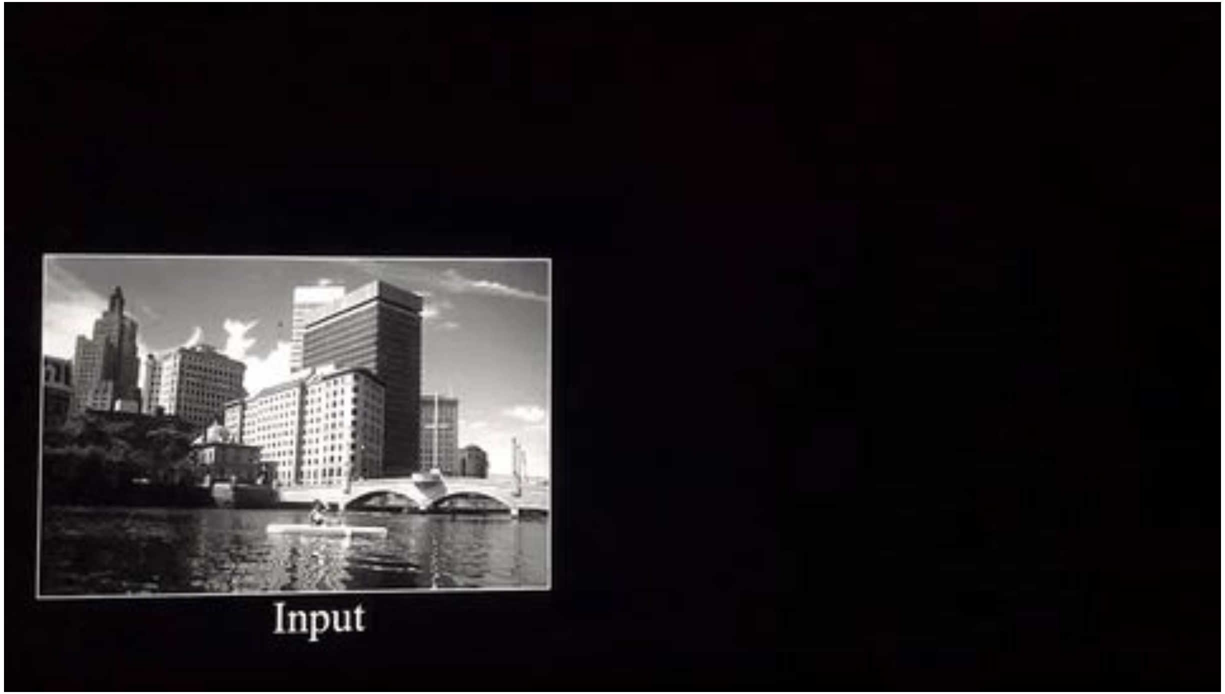
$$* \quad \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \quad =$$

Sharpen

$$* \quad \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad =$$

Convolutions - visualization

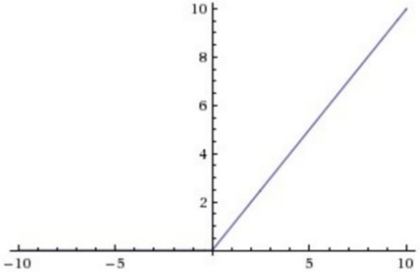
- instead of manually hand-crafting a specific filter, ConvNets are designed to learn them during the training process

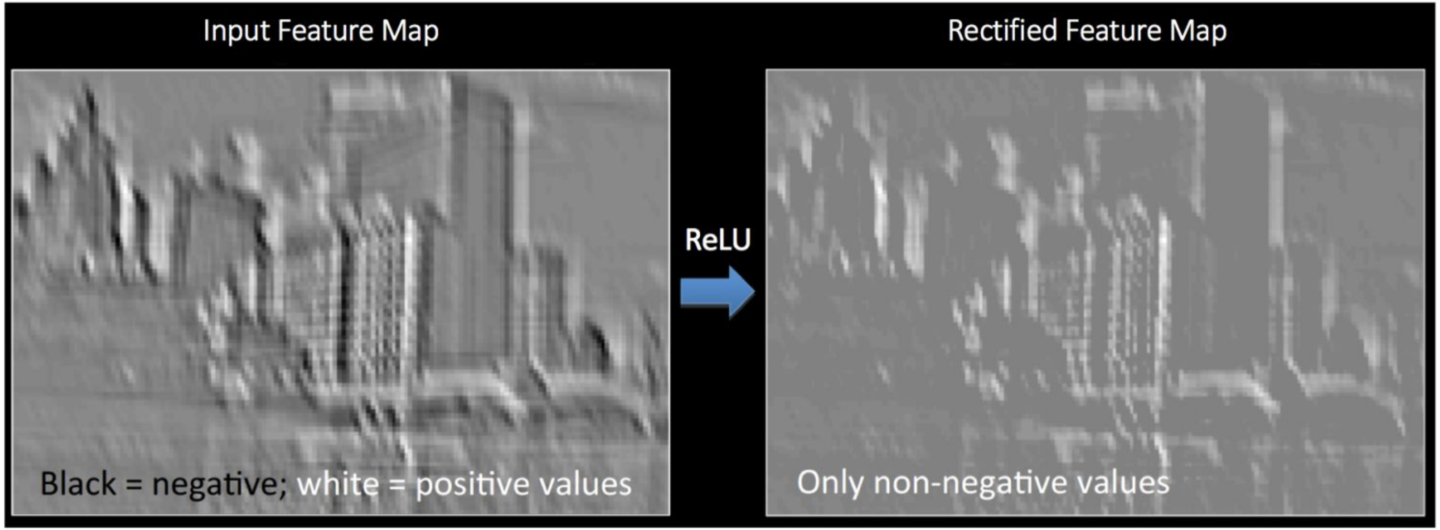


Non-linearity in ConvNets

- we add non-linear transformation after each convolution layer to make our neural network more robust and able to deal with linearly not-separable data

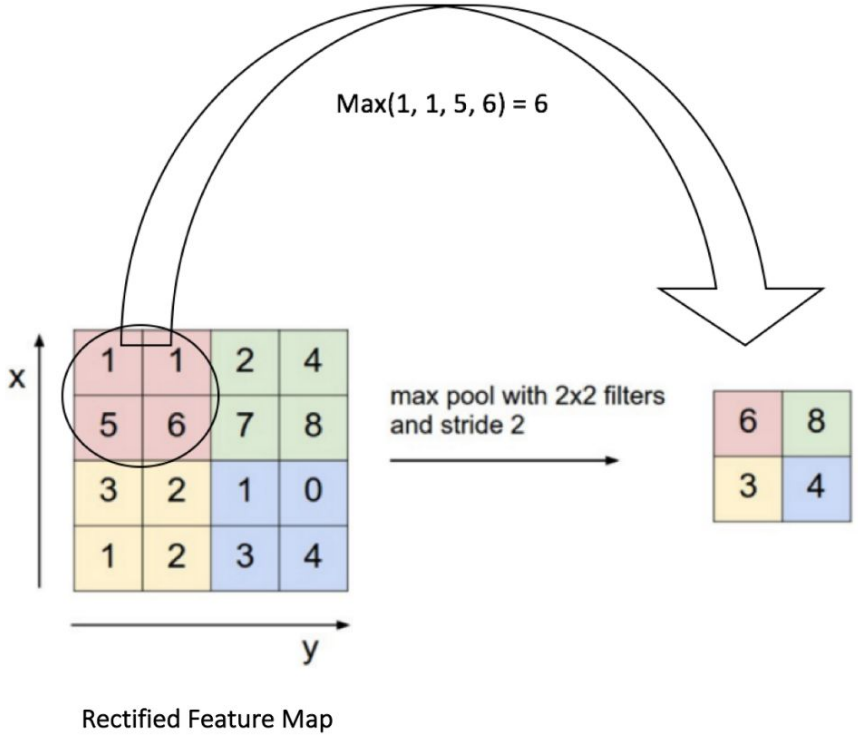
Output = Max(zero, Input)





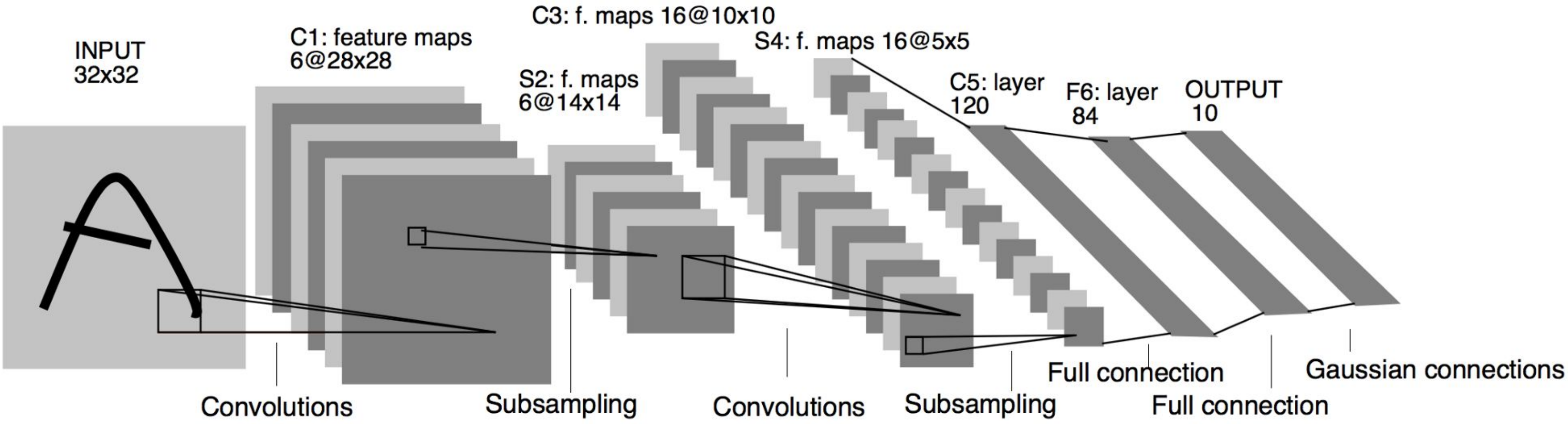
Downsampling and pooling

- **Spatial Pooling** also referred to as **subsampling** or **downsampling** reduces the dimensionality of each feature map but retains the most important information.

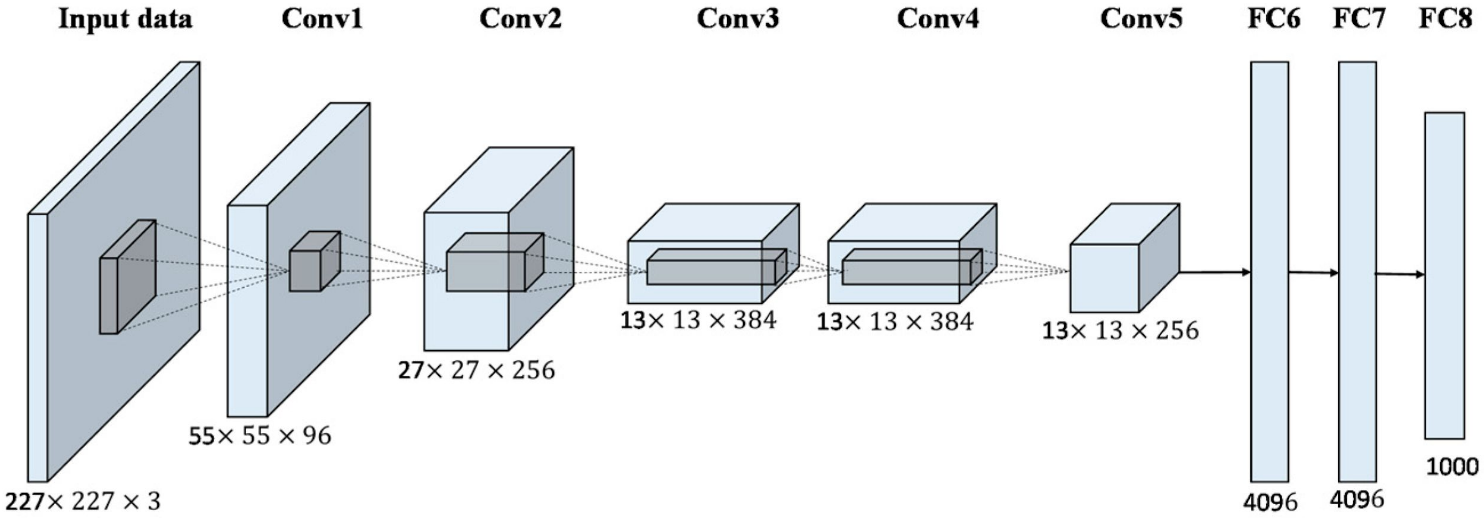


Architectures of Convolutional Neural Networks

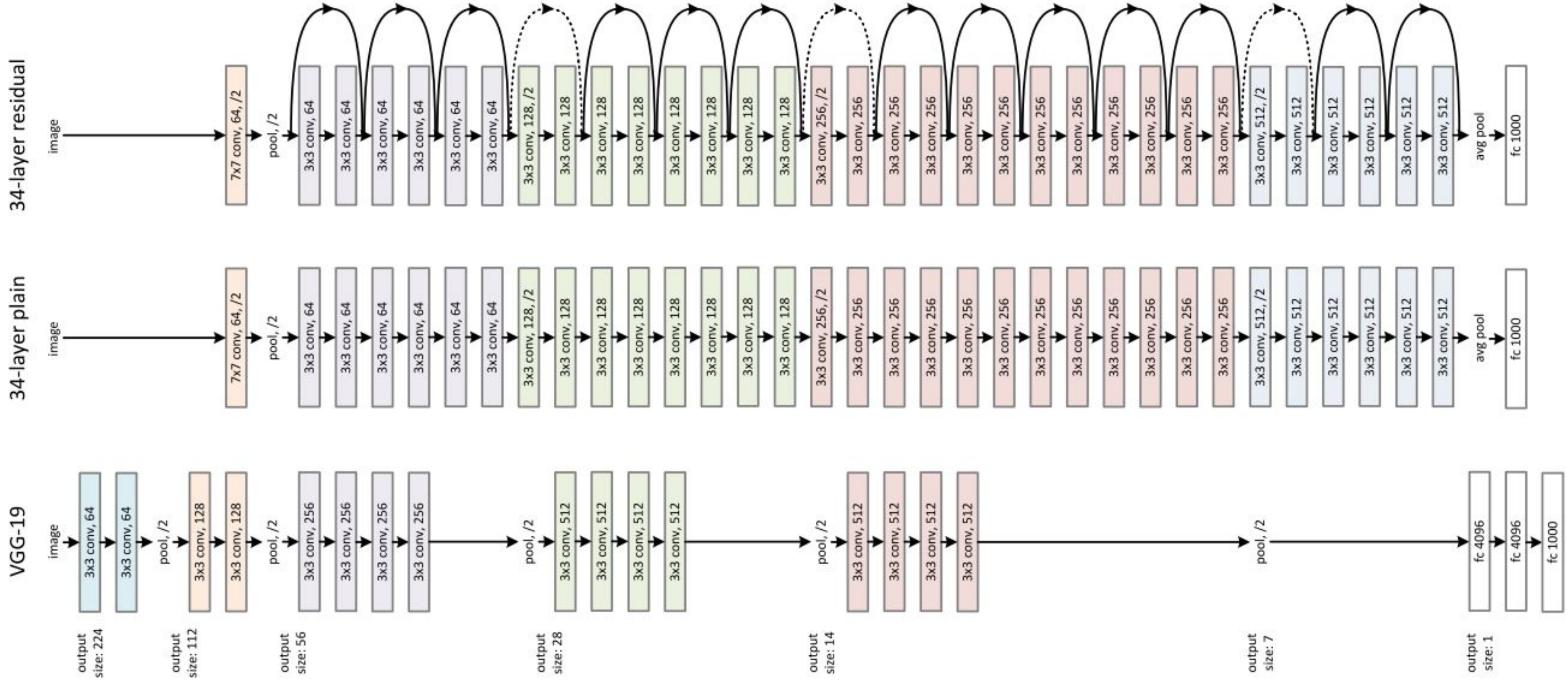
LeNet-5 architecture (1998)



AlexNet architecture (2012)



VGG-19 & ResNet-34 (2014 & 2015)

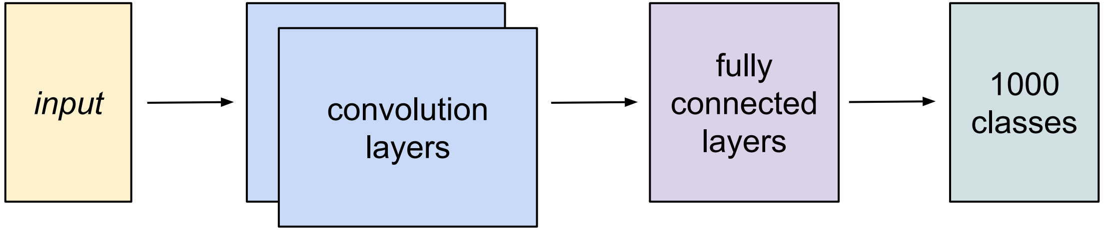


Transfer learning

Transfer learning

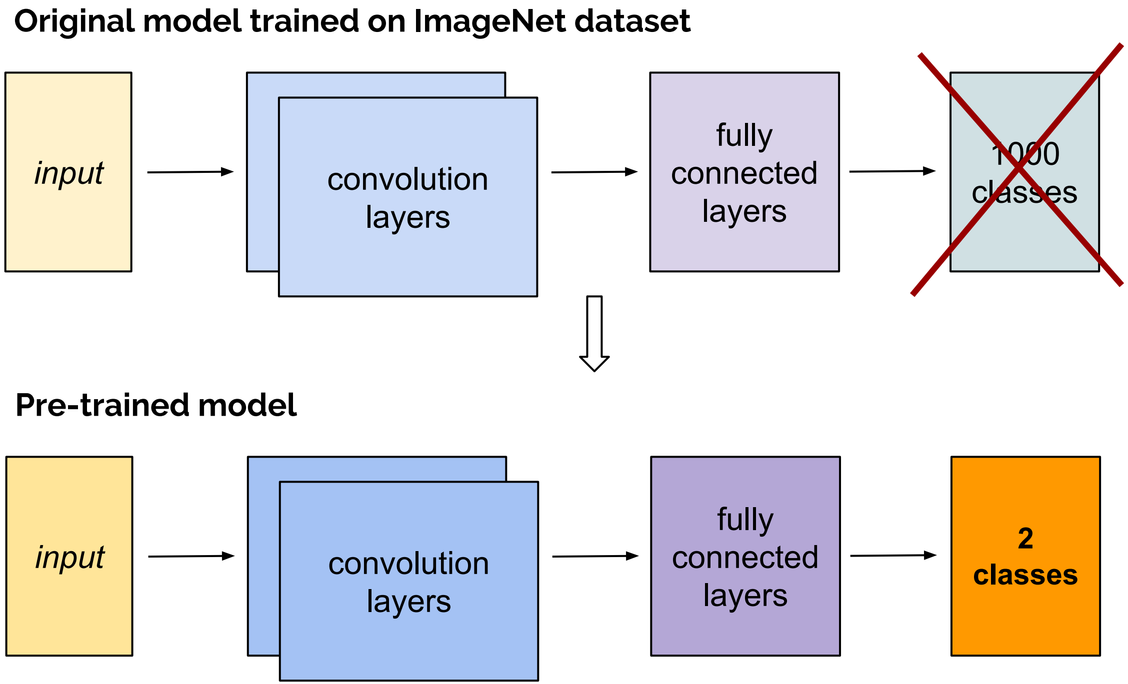
- Research problem that implies **re-using knowledge** gained while solving one problem to a **different task**.
- Example: re-use weights of the neural network learned while categorizing dogs to categorize cats.

Original model trained on ImageNet dataset



Transfer learning

- Research problem that implies **re-using knowledge** gained while solving one problem to a **different task**.
- Example: re-use weights of the neural network learned while categorizing dogs to categorize cats.



Use cases of Convolutional Neural Networks

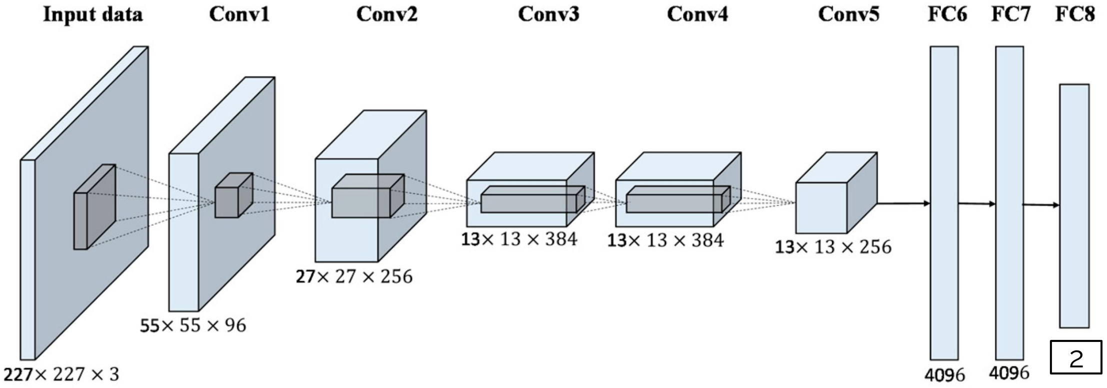
Use case: image quality assessment

- Make sure that images that are uploaded to your platform have a good quality and comply with your requirements

Good

Bad

How it works: image quality assessment



1. Label small dataset **manually**, splitting images into two classes: **good** and **bad**.

2. Use the **pre-trained ConvNet** on **ImageNet** (e.g. **ResNet34**).

3. **Train** image classifier using your own labeled dataset.

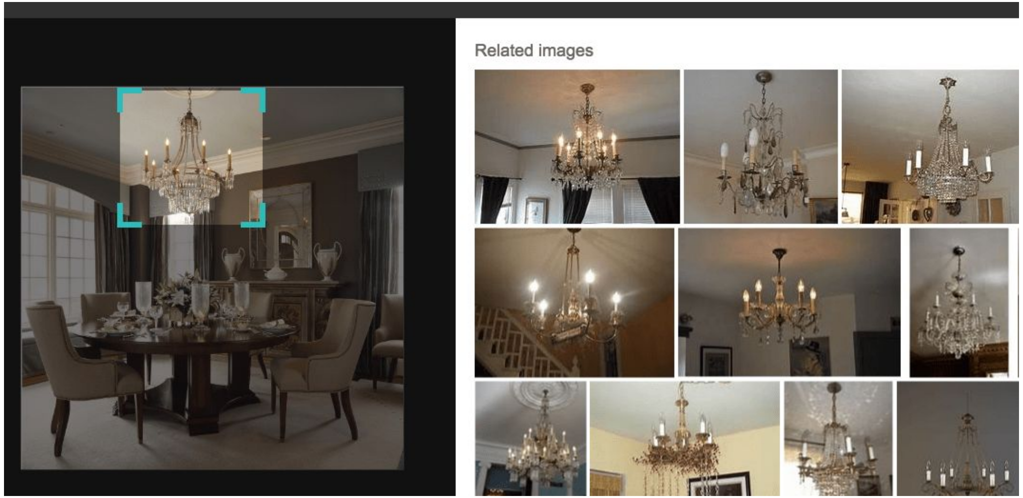
4. Once training is done - run **inference** steps on the **unlabeled dataset**.

5. **Manually** go through the newly labeled dataset, **pick misclassified** images and **add** them to the **training dataset**.

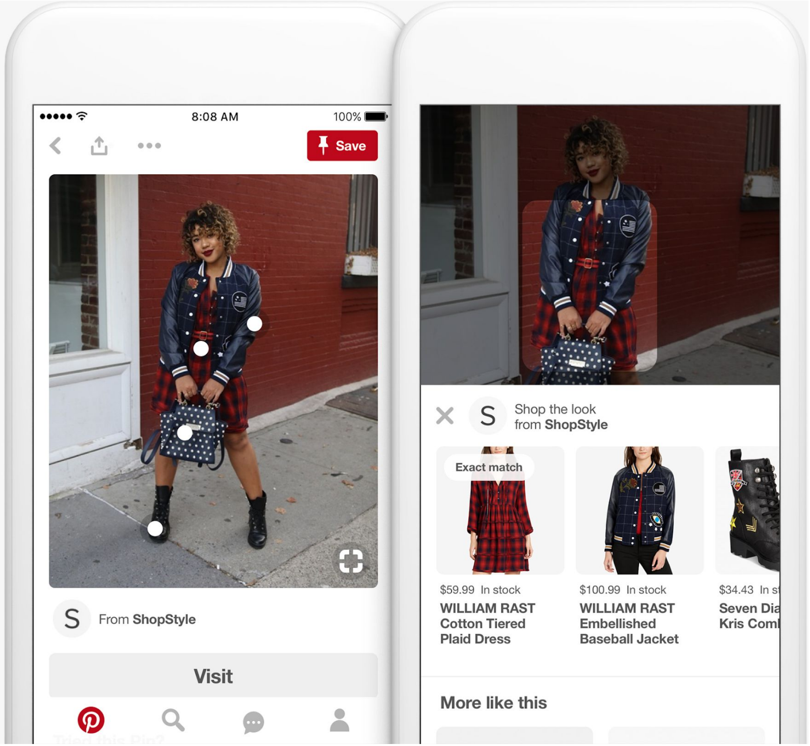
6. Go to **step 3** and **repeat**.

Use case: visual search and image similarity

- Allow the user to search products based on the photo or screenshot
- Find duplicate images in the database of e-commerce store

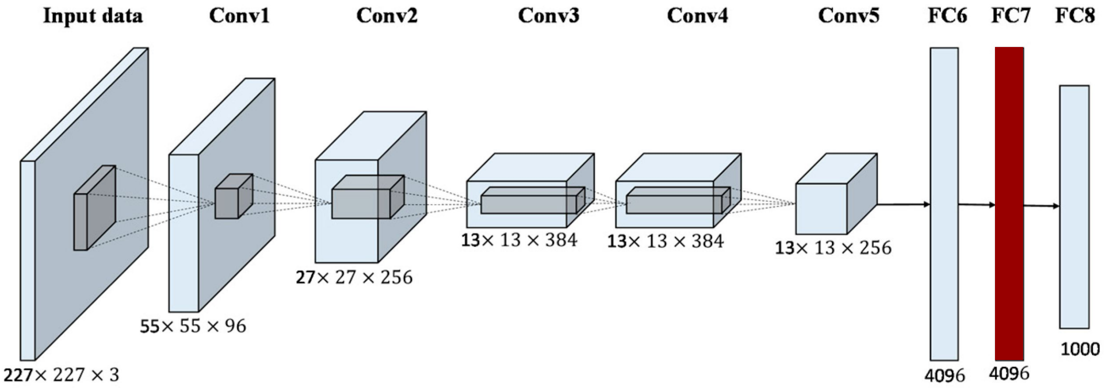


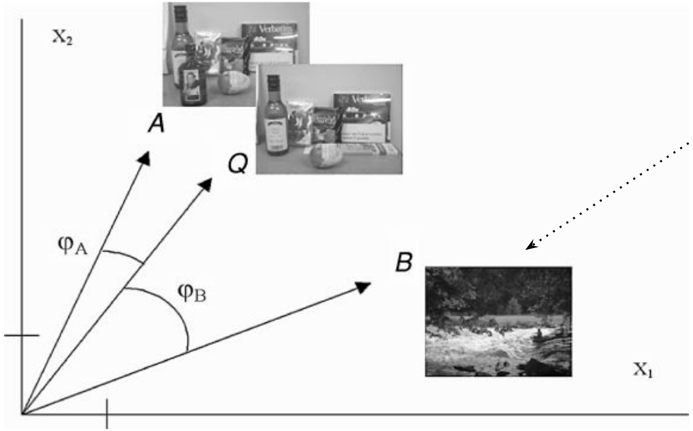
Bing



Pinterest

How it works: visual search





1. Train **ConvNet** on the original dataset to **classify** images into classes.

2. Once training is done - run **inference** step for each image in the dataset and extract it's **embedding**.

3. Last **FC layers** are typically used for image **embeddings**.

4. Compute **cosine similarity** between two embeddings (vectors) to estimate their **similarity**.

Is there a better way to implement visual search?

Visual search: advanced

- Triplet loss is minimizing a distance between an anchor and a positive sample (as they have the same identity) and maximizes a distance between an anchor and negative sample (as they have a different identity)

Anchor
a

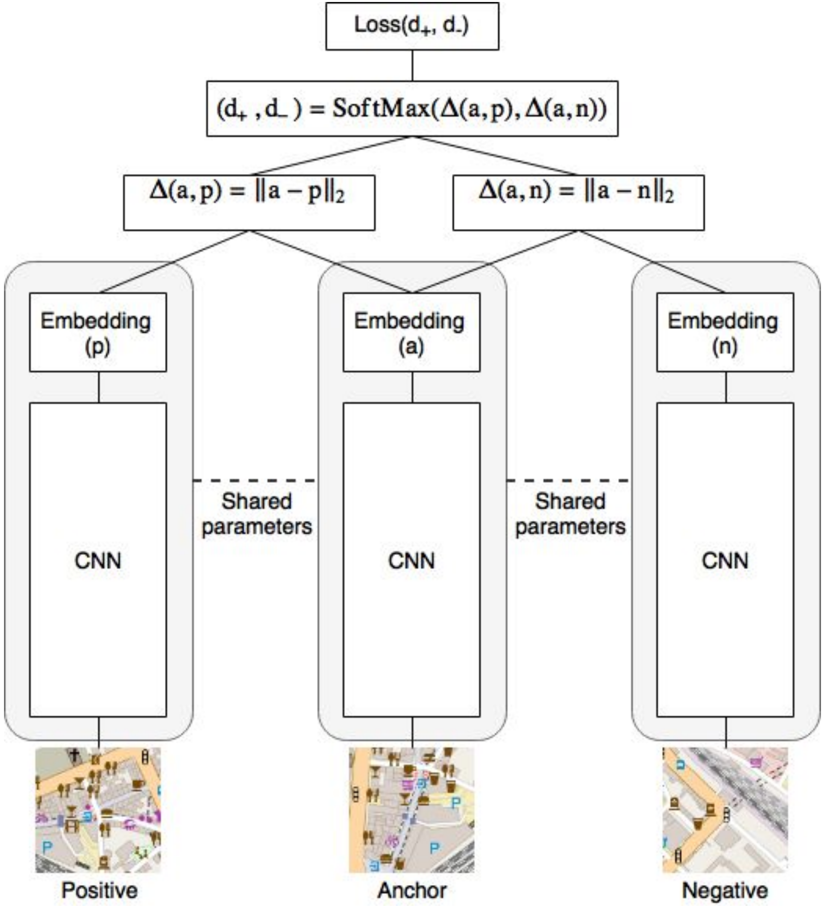
Positive
p

Negative
n

$$\mathcal{L} = max(d(a,p) - d(a,n) + margin, 0)$$

triplet loss function

Visual search: triplet loss



Code examples

https://github.com/dubai-open-data-science/deep-learning-course

PYTORCH

Question?

Thanks!