Distance, classification and learning

From Euclidean distance to deep learning

Image Feature Extraction Techniques

Piotr Fulmański

FACULTY OF MATHEMATICS
AND COMPUTER SCIENCE
University of Lodz

Remark

Although the main goal of this lecture is to give some tools you can use to transform images to more useful form (extract some feature or create image/feature descriptors) suitable for further processing I will discuss very popular image processing tool which are neural networks.

Main reason for this is that deep neural networks (DNN), especially one of their branch, so called convolutional neural networks (CNN) extract features in their hidden layers.

Other words, they unify the process of feature extraction, data (image) processing and learning from examples.

Distance and classification

Distance and classification

The k-nearest neighbour for classification

The k-nearest neighbour for classification

Neural networks

Neural networks

From previous part you know that classification is the process by which we assign class labels to sets of measurements.

Classification depends on *distance* and/or *structure*.

The latter defines the heart of pattern recognition: *can we **learn** the structure of the data*?

Neural networks

One possible approach, which is a common alternative to using a (statistical) classification rule, is to use a ***neural networks*** (known as Artificial Neural Networks (ANNs)) in which the processing elements aim to mimic properties of neurons in the human brain.

Modern approaches focus on a multilayer perceptrons (MLPs), where the input information is transformed to a *classification decision* via a hidden layer (layer which outputs are not the classification outputs).

The universal approximation theorem states you need only one hidden layer to approximate an arbitrary continuous function. [6, 7]

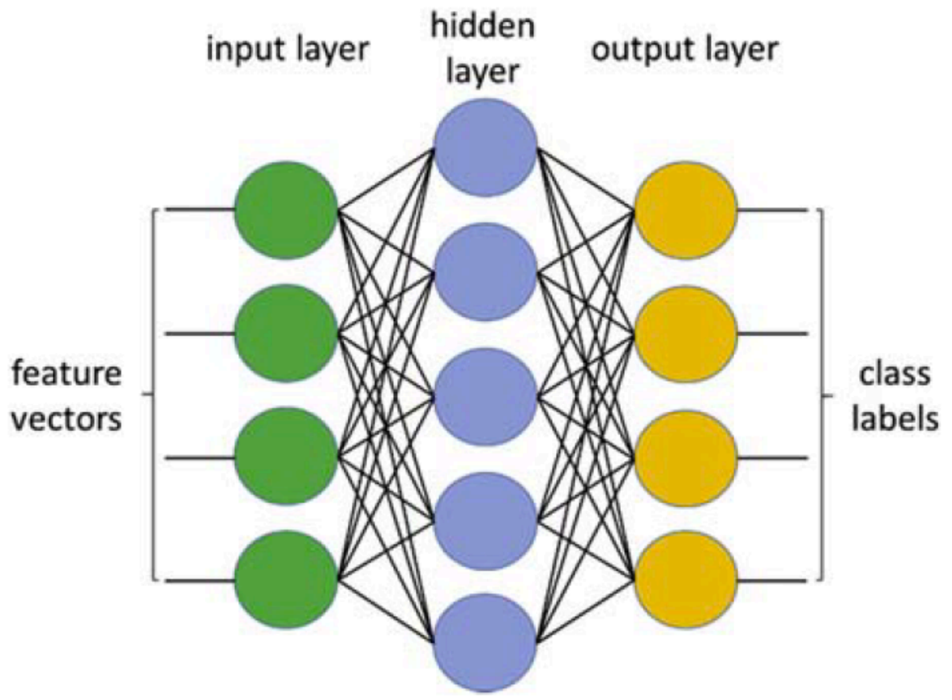


Image source: [1]

Neural networks

The function of the artificial neural network is to **learn** the class labels from feature vectors using *computational units — neurons*.

In basic form the neurons form a function f of the sum of the inputs x multiplied by their respective weights w.

The networks constructed from these neurons require **training**, typically by error *back-propagation.* During the learning process, the weights change aimed to minimise a *loss function* thereby minimising classification error on the training data.

When training ends, the network should have learnt how to recognise the data. We believe it learned data structure and the output of a neural network can be arranged to be class labels.

What should be stressed, learning proces is:

- **highly random,**
- **very nondeterministic,**
- **results very often are not repeatable.**

Support Vector Machines (SVMs)

Support Vector Machines (SVMs)

Support vector machines have nothing in common with machines, but rather with a set of equations designed to give an optimal classification result in a **repeatable and controlled manner**.

Originally, linear SVMs aimed to define a *hyperplane* that could separate classes. The best hyperplane was that which maximised the margin between the two classes, with support vectors positioned on the edges between the feature clusters.

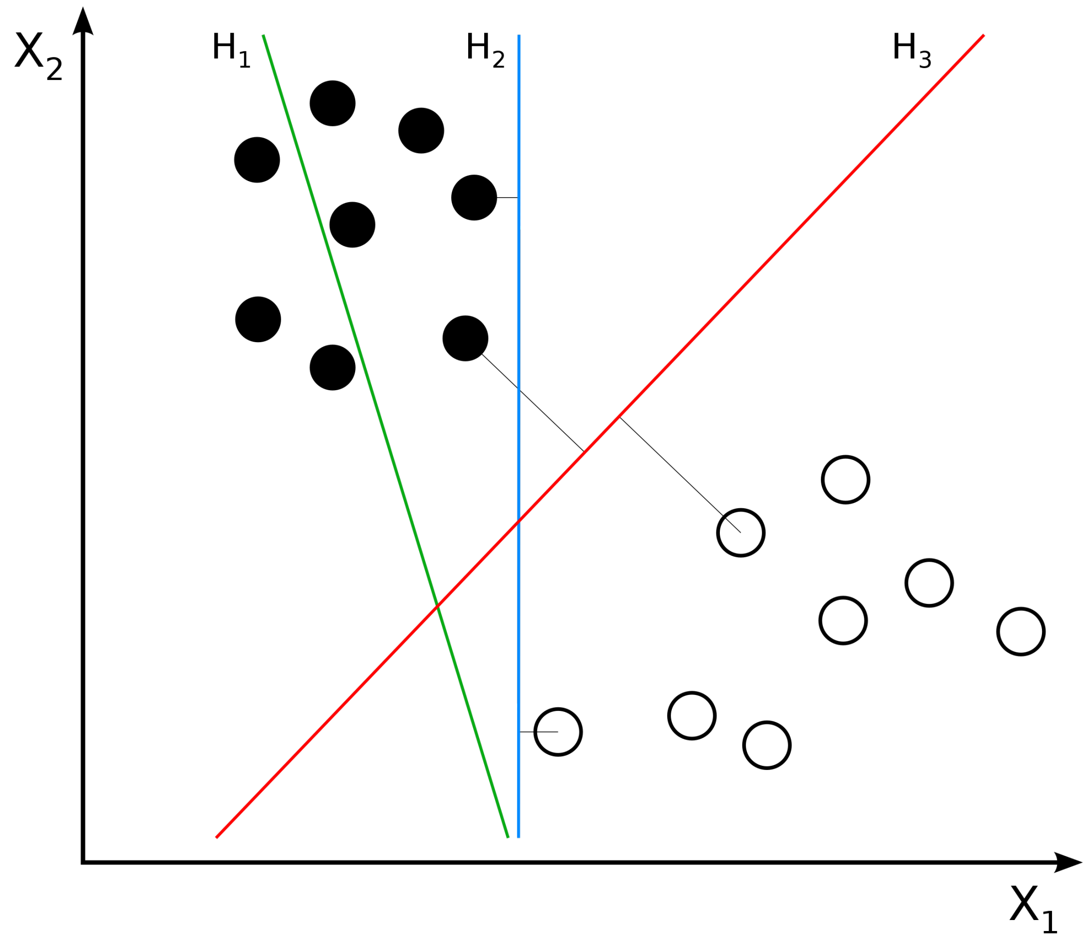


Image source: [2]

Support Vector Machines (SVMs)

Maximum-margin hyperplane and margins for an SVM trained with samples from two classes. Samples on the margin are called the **support vectors**.

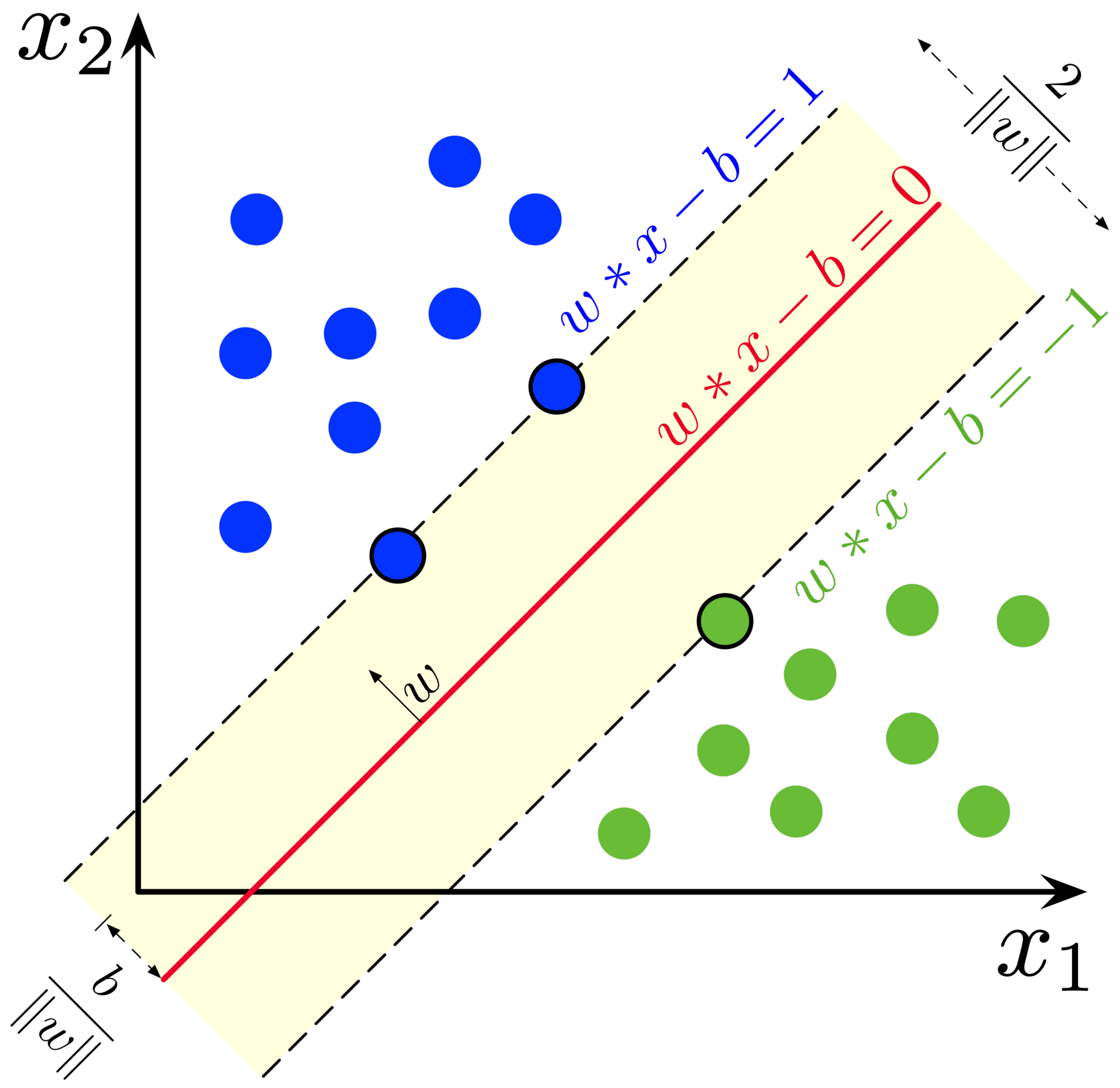


Image source: [2]

Support Vector Machines (SVMs)

It often happens that the sets to discriminate are not linearly separable in that space. For this reason, it was proposed that the **original finite-dimensional space be mapped into a much higher-dimensional space**, presumably making the separation easier in that space, where linear separation could be used.

To keep the computational load reasonable, the mappings used by SVM schemes are designed to ensure that dot products of pairs of input data vectors may be computed easily in terms of the variables in the original space, by defining them in terms of a kernel function $k(x, y)$. This method is known under the name *kernel trick* (see [3] for examples, [4] for deep mathematical concepts, [5] to understand SVM).

Support Vector Machines (SVMs)

The resulting algorithm is formally similar, except that every dot product is replaced by a nonlinear kernel function. **In result, although the classifier is a hyperplane in the transformed feature space, it may be nonlinear in the original input space.**

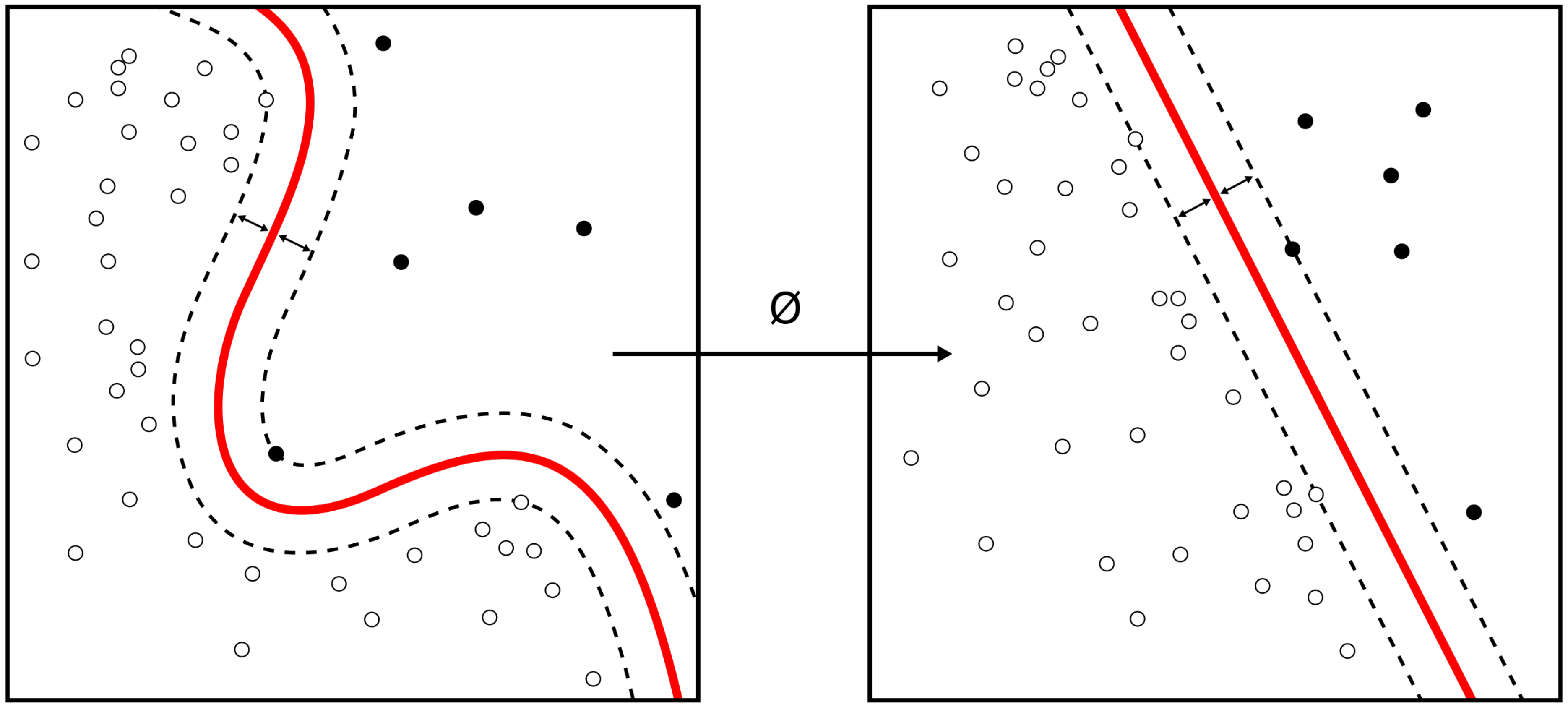


Image source: [2]

Deep learning

Ideas

Deep learning

So until the deep learning era, feature extraction and learning from data were separate. Deep learning joined them together.

Deep learning

The origins of deep learning are some time ago now when **it was shown that multilayer neural networks could achieve good results** by gradient-based learning to minimise classification error (or loss).

Unfortunately, **the training time was long, with many parameters and little data**.

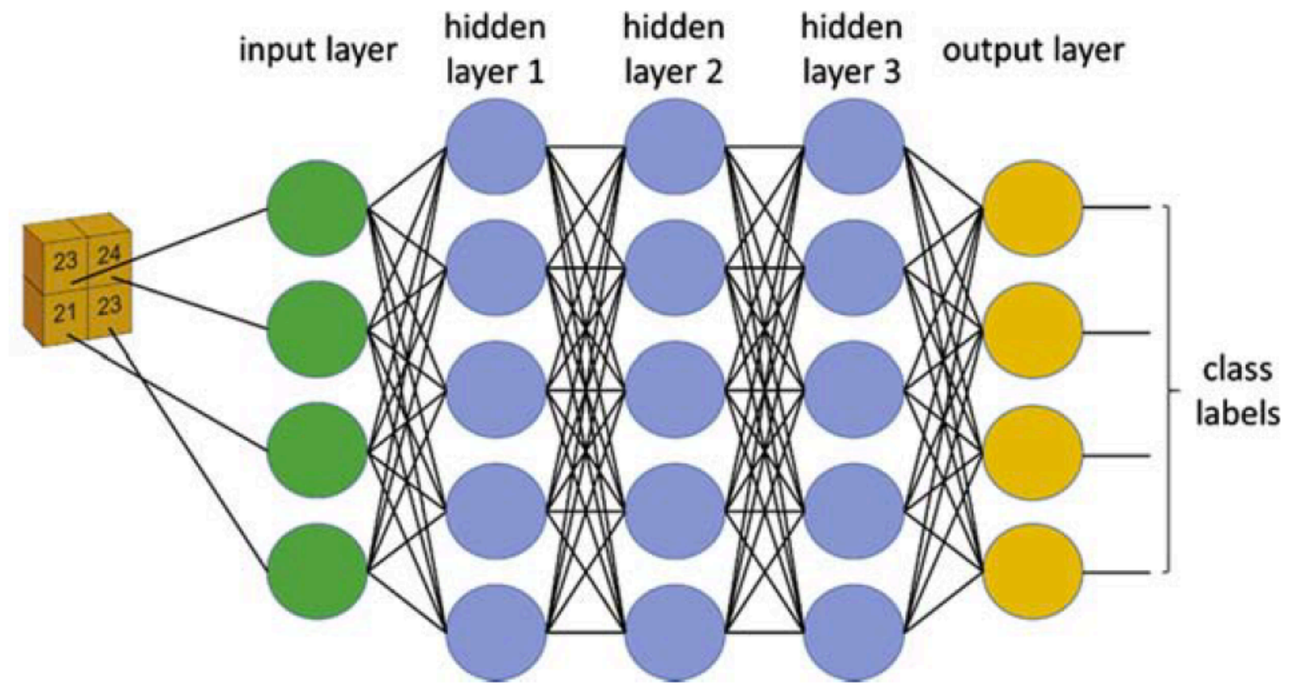
Later the **hardware advanced rapidly** in terms of faster processing speeds (especially using graphics processing units (GPUs)) with cheaper and larger memory and so the networks could become larger.

It was possible because working with neural networks is nothing else like matrix multiplication — an operation for which GPSu are a perfect tool.

They were shown to have excellent performance on an image classification problem and that is where the revolution started.

Deep learning

Below you have the basic architecture of a deep neural network:



The **input data are an image** and the **output is class labels**.

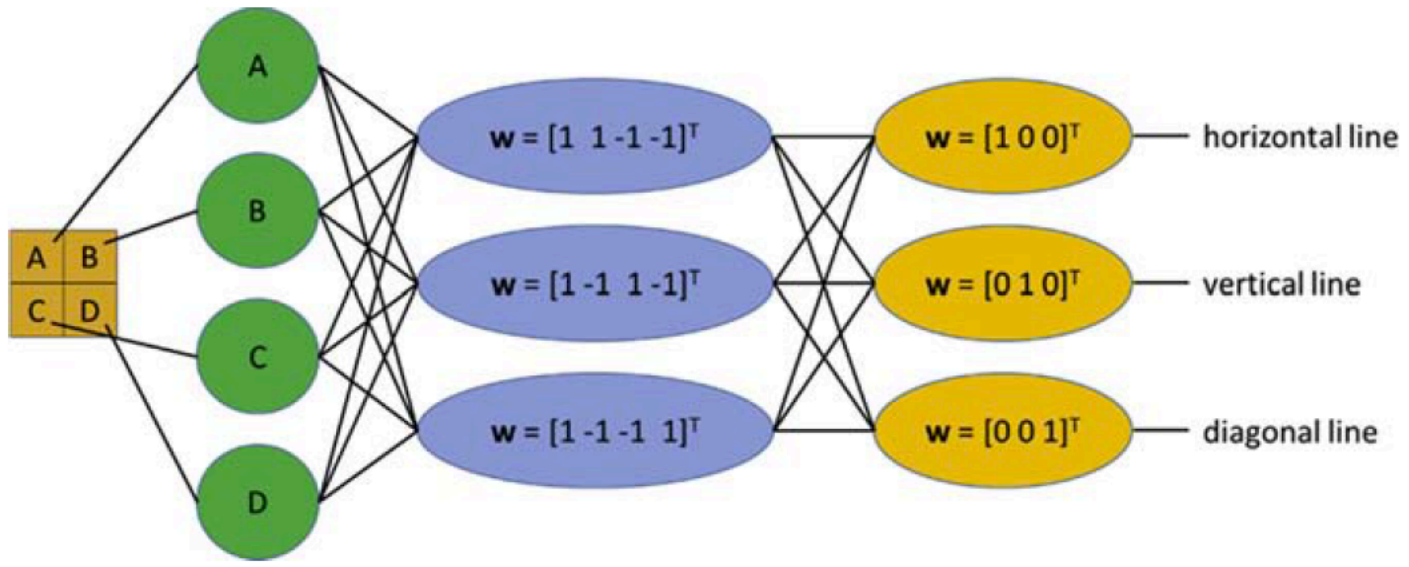
Feature extraction is performed within the network to achieve the classification decision.

To do this efficiently and reliably, it needs (much) **more layers** than the earlier ANN, **thus it is a deep network**.

Image source: [1]

Deep learning

If, for example, you arrange an architecture to detect horizontal, vertical and diagonal lines in a 2 x 2 image, you can use an architecture of the form shown below:



After training the network on a set of images you can obtain the appropriate weights to detect the given classes of shape in the image.

Here you combine feature detection and analysis with classification decisions.

Image source: [1]

Deep learning

Examples of the resulting outputs of this network are shown below:

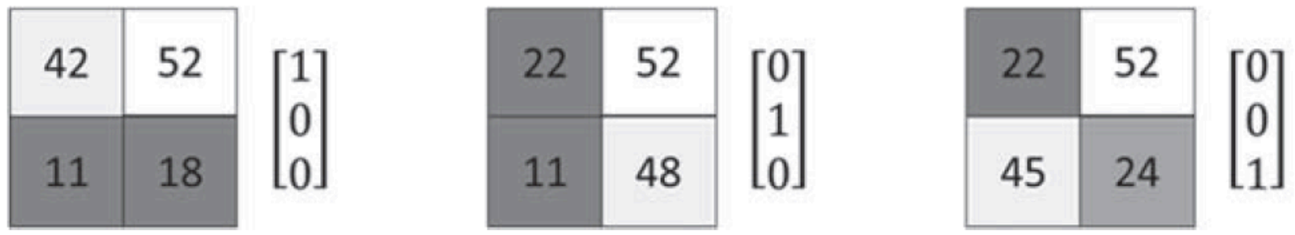


Image source: [1]

Deep learning

The deep learning revolution was initiated by Alexnet. According to Wikipedia (https://en.wikipedia.org/wiki/AlexNet):

AlexNet is considered one of the most influential papers published in computer vision, having spurred many more papers published employing CNNs and GPUs to accelerate deep learning. As of 2021, the AlexNet paper has been cited over 80,000 times according to Google Scholar.

It achieved markedly better performance than any other technique on a competition called the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC, https://image-net.org) that had labelled high-resolution images from around 1000 categories giving 1.2 million training images, 50,000 validation images and 150,000 images for testing. Alexnet achieved the lowest error and simply blew away the opposition.

Note: In 2015, AlexNet was outperformed by Microsoft's very deep CNN with over 100 layers, which won the ImageNet 2015 contest. Researchers at Microsoft reported that their CNNs exceeded human ability at the narrow ILSVRC tasks. However, as one of the challenge's organizers, Olga Russakovsky, pointed out in 2015, the programs only have to identify images as belonging to one of a thousand categories; humans can recognize a larger number of categories, and also (unlike the programs) can judge the context of an image.

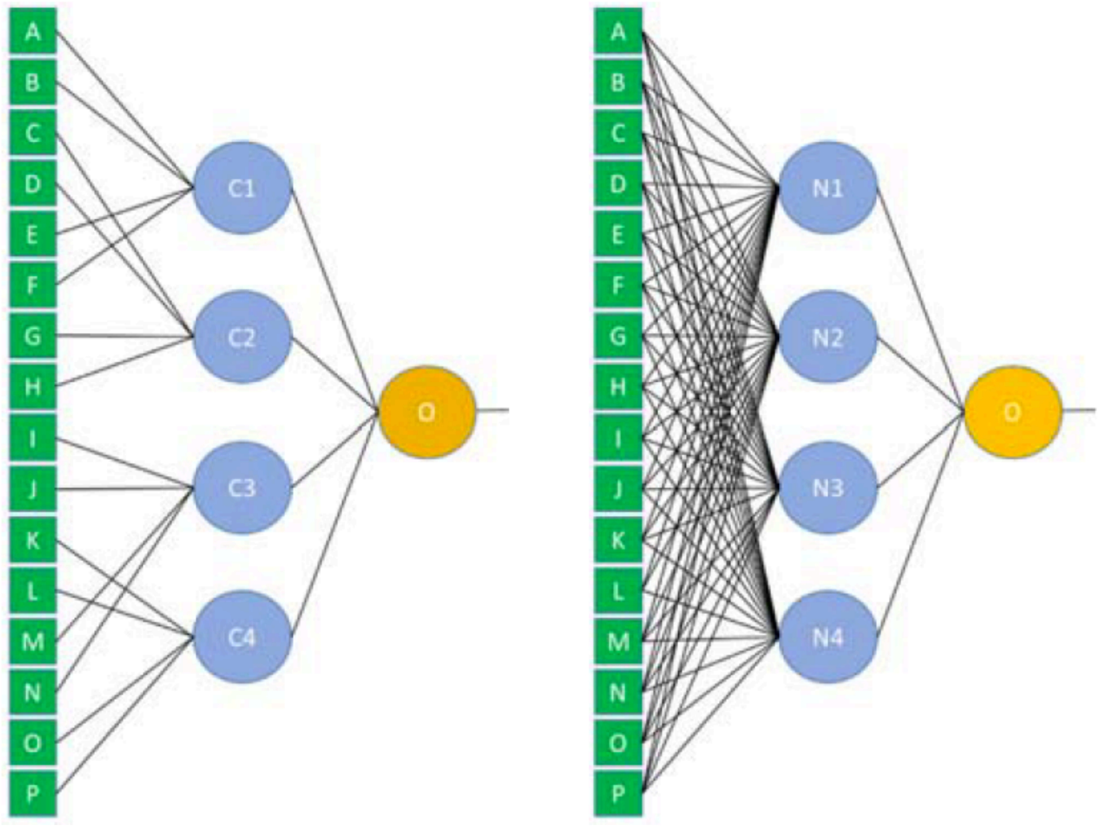
Computation in Alexnet uses layers of 650,000 neurons and 60 million parameters with convolutional layers and fully connected ones. The convolutional layers implement neural structures that operate the area function in convolution, which you saw earlier. One main concept in convolution was to use local structure

Deep learning

Main building block of this network are convolution layers.

The convolutional architecture considers local regions within an image, whereas the MLP looks at everything.

The difference between a convolutional layer and an MLP is shown below:



As you can see, there are many parameters to be learnt in the MLP, and comparatively many fewer in the convolutional network.

Image source: [1]

Deep learning

In the past, **traditional multilayer perceptron** (MLP) models were used for image recognition. However, **the full connectivity between nodes caused the curse of dimensionality**, and was computationally intractable with higher resolution images.

For example:

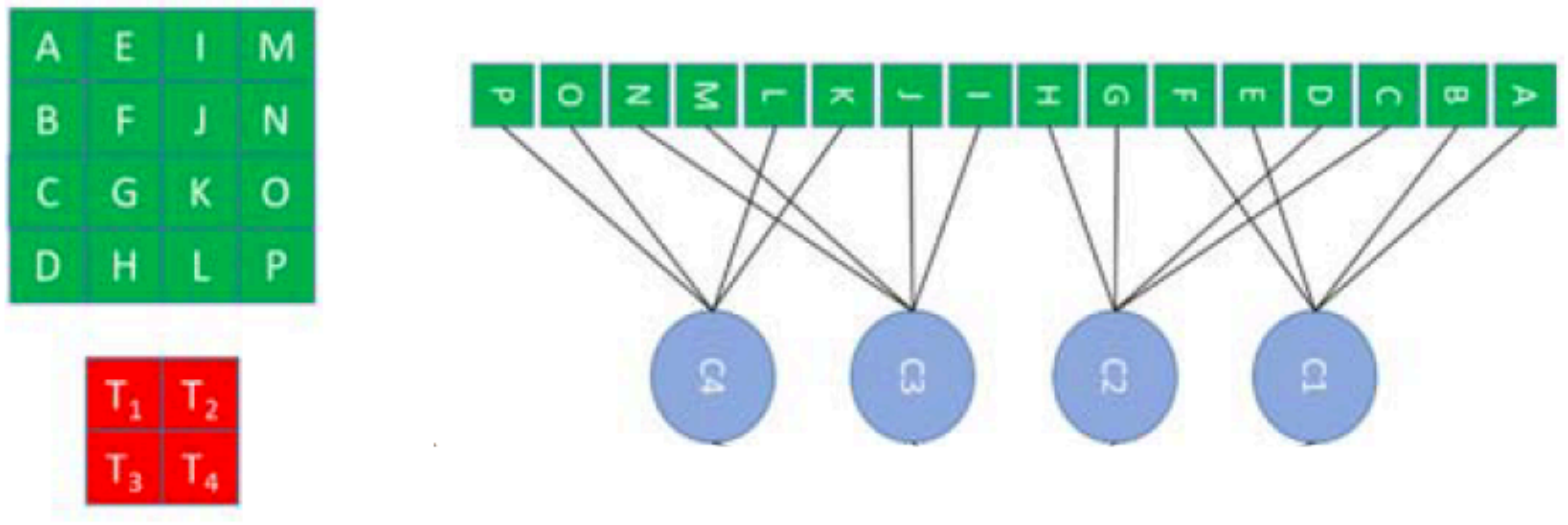
- In CIFAR-10, images are only of size 32 x 32 x 3 (32 wide, 32 high, 3 color channels), so a single fully connected neuron in the first hidden layer of a regular neural network would have 32 * 32 * 3 = 3072 weights.

- A 200 x 200 image, however, would lead to neurons that have 200 * 200 * 3 = 120000 weights.

- A 1000 x 1000-pixel image with RGB color channels has 3 million weights, which is too high to feasibly process efficiently at scale with full connectivity.

Also, **typical MLP network architecture does not take into account the spatial structure of data**, treating input pixels which are far apart in the same way as pixels that are close together. This ignores locality of reference in data with a grid-topology (such as images), both computationally and semantically. Thus, full connectivity of neurons is wasteful for purposes such as image recognition that are dominated by spatially local input patterns.

Deep learning



The layer's parameters consist of a set of learnable filters (or kernels) — red squares above, which have a small receptive field, but extend through the full depth of the input volume — green squares above.

During the forward pass, each filter is convolved across the width and height of the input volume, computing the dot product between the filter entries and the input, **producing a 2-dimensional activation map of that filter**.

As a result, the network learns filters that activate when it detects some specific type of feature at some spatial position in the input.

Stacking the activation maps for all filters along the depth dimension forms the full output volume of the convolution layer.

Every entry in the output volume can thus also be interpreted as an output of a neuron that looks at a small region in the input and shares parameters with neurons in the same activation map.

Image source: [1]

Deep learning

Typical CNN architecture:

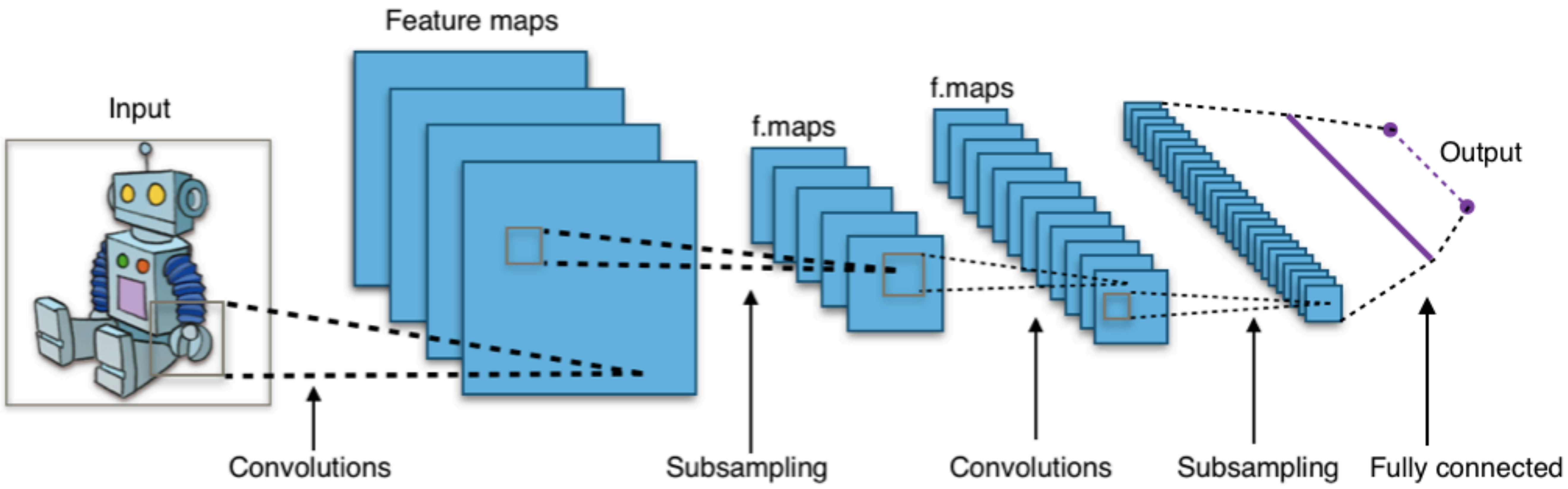


Image source: https://en.wikipedia.org/wiki/Convolutional_neural_network

Deep learning

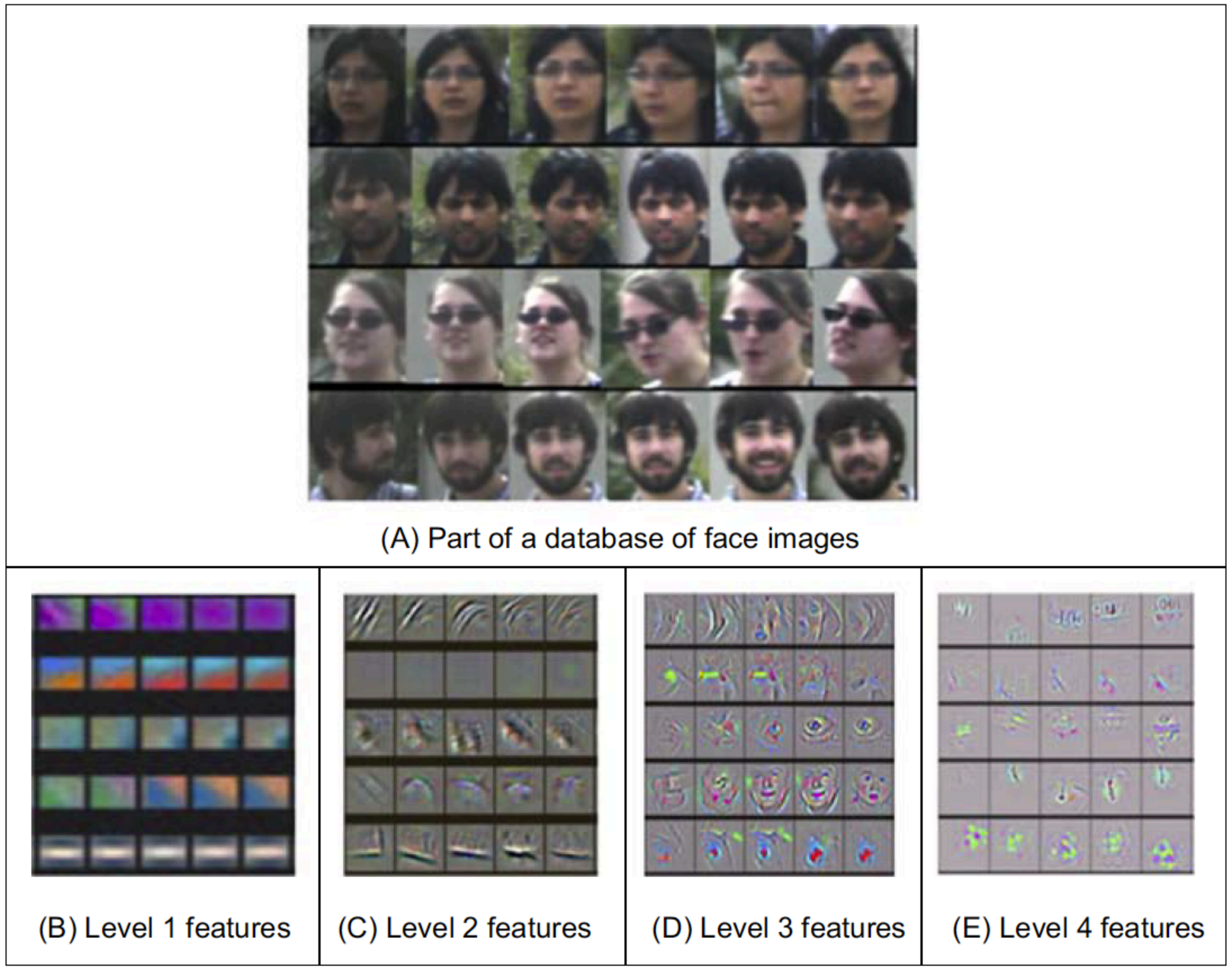


Image source: [1]

Deep learning

The convolution layers of a CNN have neurons arranged in 3 dimensions: width, height and depth. Where each neuron inside a convolutional layer is connected to only a small region of the layer before it, called a receptive field.

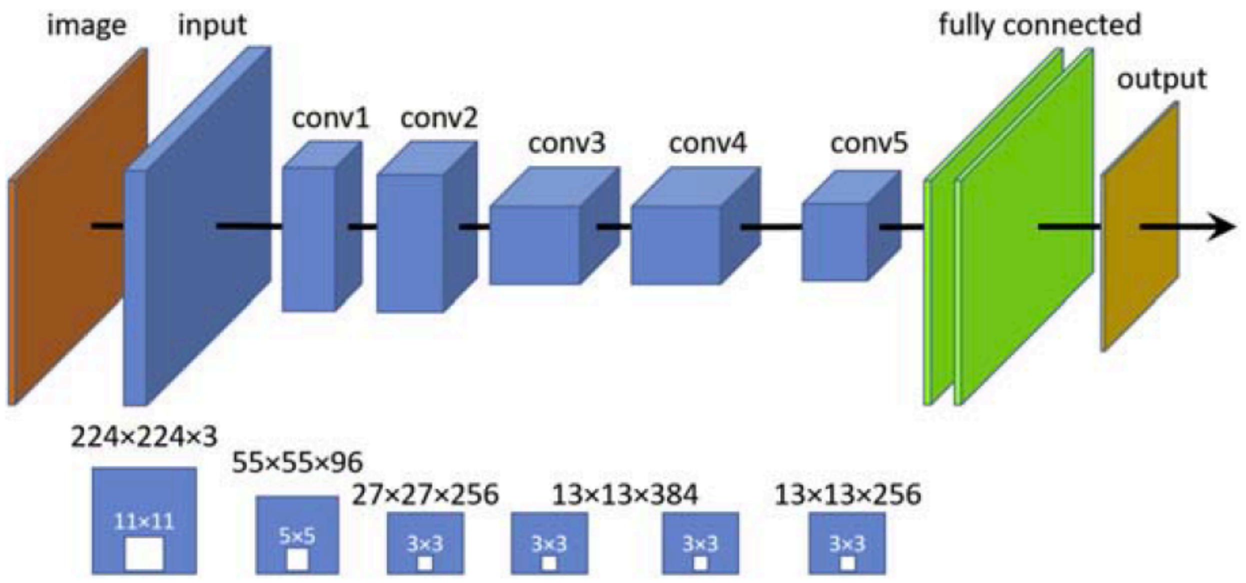
The size of the output volume of the convolutional layer is controlled by:

- The **depth** of the output volume controls the number of neurons in a layer that connect to the same region of the input volume. These neurons learn to activate for different features in the input. For example, if the first convolutional layer takes the raw image as input, then different neurons along the depth dimension may activate in the presence of various oriented edges, or blobs of color.

- **Stride** controls how depth columns around the width and height are allocated. If the stride is 1, then we move the filters one pixel at a time. This leads to heavily overlapping receptive fields between the columns, and to large output volumes. For any integer $s > 0$ a stride s means that the filter is translated s units at a time per output. A greater stride means smaller overlap of receptive fields and smaller spatial dimensions of the output volume.

Deep learning

Having indispensable knowledge about convolutional layer, you are ready to see AlexNet architecture:



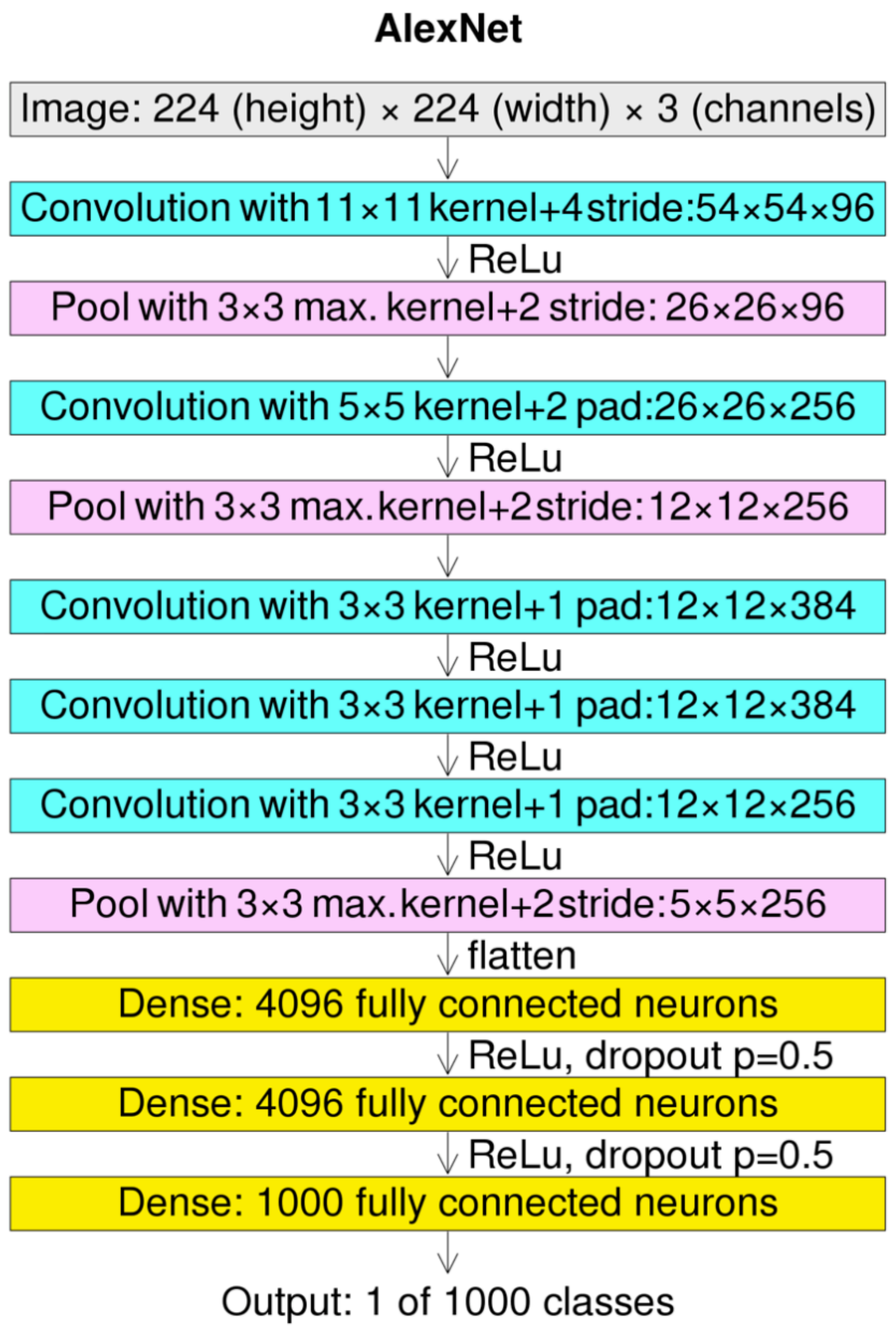
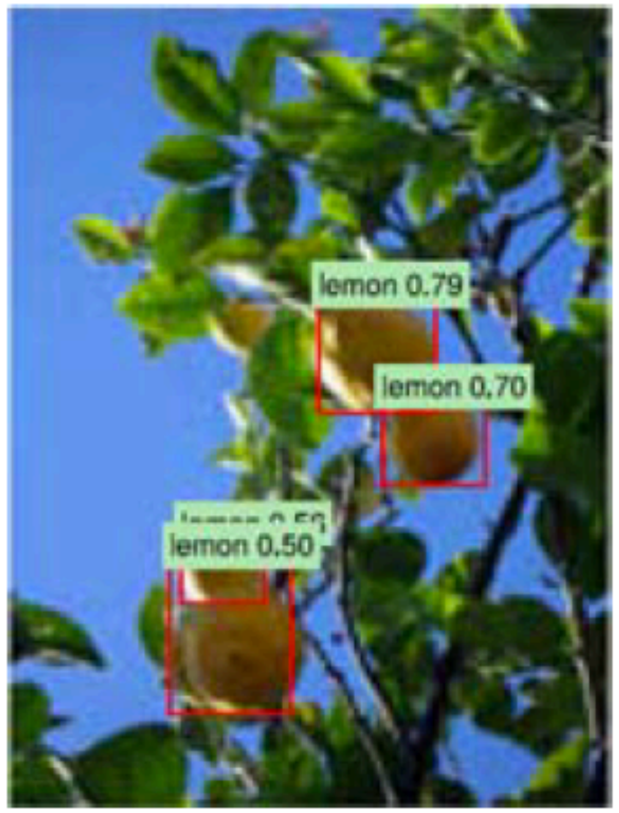
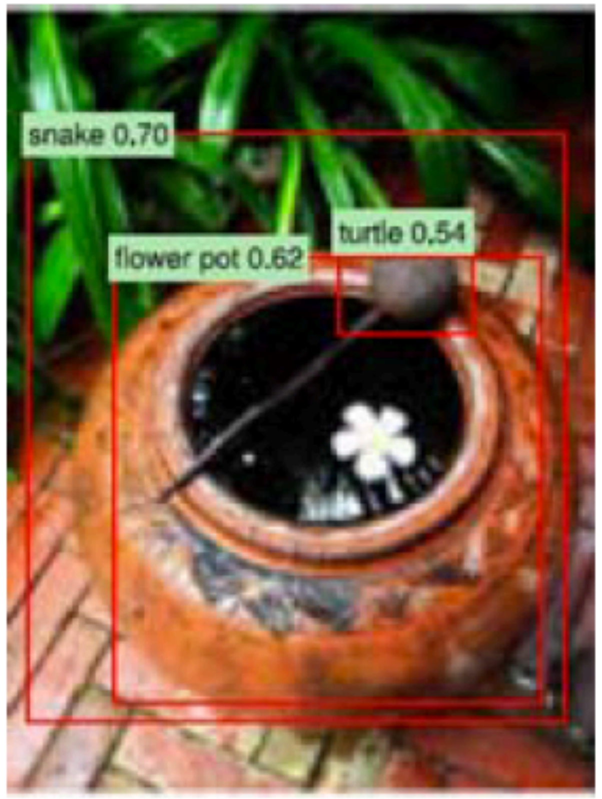

AlexNet
Image: 224 (height) × 224 (width) × 3 (channels)
Convolution with 11×11 kernel+4 stride:54×54×96
ReLu
Pool with 3×3 max. kernel+2 stride: 26×26×96
Convolution with 5×5 kernel+2 pad:26×26×256
ReLu
Pool with 3×3 max.kernel+2stride:12×12×256
Convolution with 3×3 kernel+1 pad:12×12×384
ReLu
Convolution with 3×3 kernel+1 pad:12×12×384
ReLu
Convolution with 3×3 kernel+1 pad:12×12×256
ReLu
Pool with 3×3 max.kernel+2stride:5×5×256
flatten
Dense: 4096 fully connected neurons
ReLu, dropout p=0.5
Dense: 4096 fully connected neurons
ReLu, dropout p=0.5
Dense: 1000 fully connected neurons
Output: 1 of 1000 classes

Image source: [1], https://en.wikipedia.org/wiki/AlexNet

Deep learning

Applications





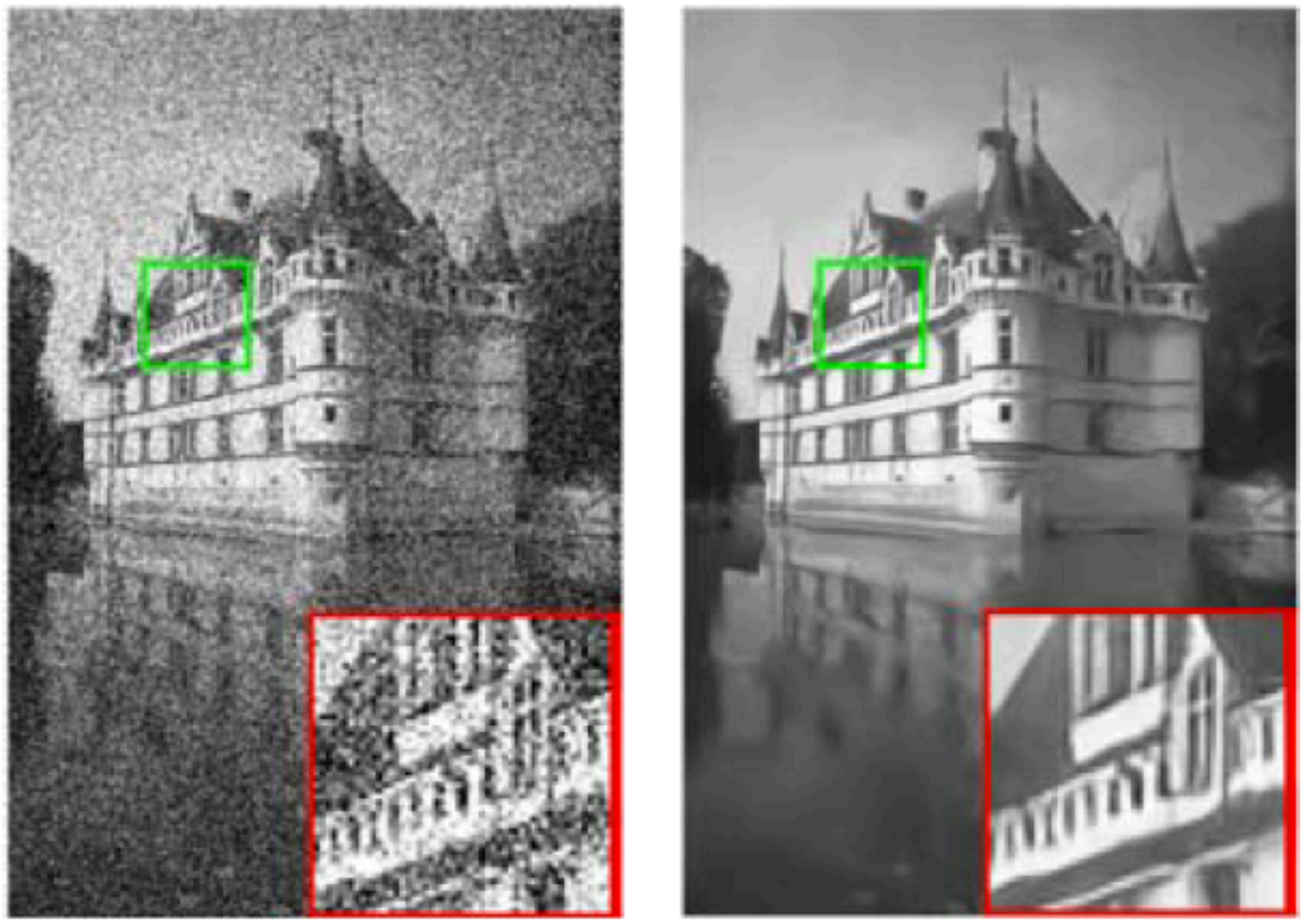


Image source: [1]

What is wrong with deep learning

What is wrong with deep learning

Deep learning

What is wrong

Deep neural networks have achieved very high, sometimes competing with human capabilities, performance in many computer vision tasks. Based on these successes, they are increasingly being used as components of car control systems, unmanned aerial vehicles and robots. Note, however, that **deep neural networks do not add anything new to the theoretical basis of the construction and operation of artificial neural networks**.

They have been known since the use of multi-layer artificial neural networks. Their potential were also known a long time ago. The fact that they flourish in recent years is due not to any concepts or ideas that are worth attention, but to the availability of hardvare that is capable of mass matrix computing (mainly graphics cards), which allowed their use in practice.

Deep learning

What is wrong

From a mathematical point of view, **every neural network is a function**. Theoretically, one could study its properties and thus say something about the operation of a particular network. In practice, the degree of complexity of such a network (function) is so great that its analysis is an extremely complex issue.

Consequently, usually **all we know is that the input signals are "somehow" converted into output signals**. Unfortunately, we cannot analyze this process in more detail. **Thus, the artificial neural network is referred to as a "black box" that "magically" performs a specific task.** At least that's what we'd like to believe.

The main claim of artificial neural networks is that they embody new and powerful general principles of information processing. These rules are poorly defined. It is often claimed that they emerge from the web itself. Thus, a simple statistical association (a fundamental function of artificial neural networks) can be described as learning or recognition.

Deep learning

What is wrong

According to [8], in 1997, Alexander Dewdney commented on this saying that:

solutions are found as if by magic; and no one, it seems, has learned anything.

Also in [8] a technology writer Roger Bridgman commented:

you could create a successful net without understanding how it worked: the bunch of numbers that captures its behaviour would in all probability be "an opaque, unreadable table... valueless as a scientific resource".

Deep learning

What is wrong

Methodology of working with neural networks is primarily based on a series of trial and error. You try to build the right network based on *experience* and *intuition* until at some point in testing you get a satisfactorily low error rate. Then we state that the network is learned and ready to be used. And the network starts working. As an element of the controller in an airplane, car, diagnostic tool or program that decides about granting a loan for a house.

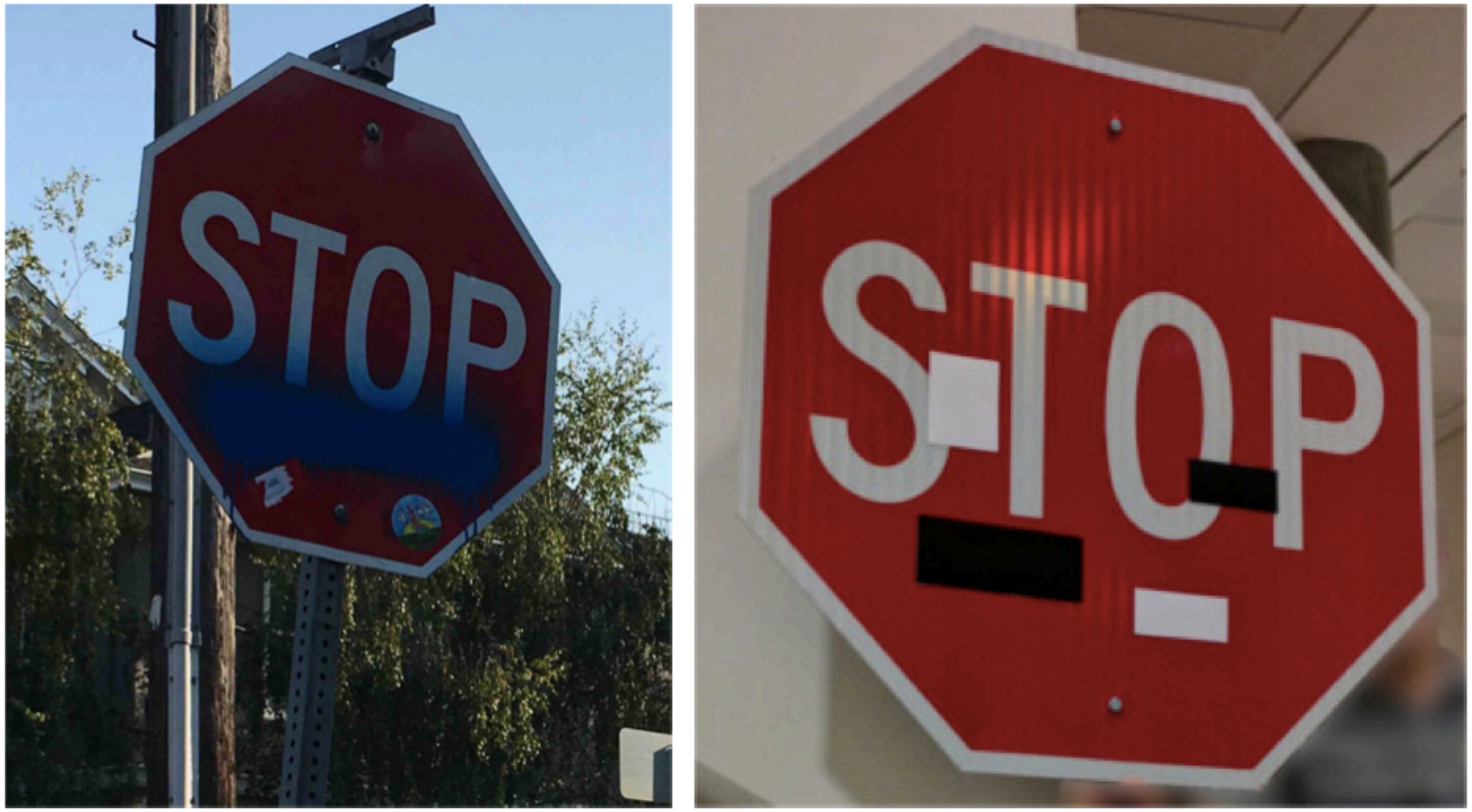
There are no methods that can "calculate" the artificial neural network. In fact, we can't say anything about the "ready" network. Maybe except stating that it works on test data.

This, however, is not any prove. People approve the truth of the model on the basis of the correctness of the model built on the basis of a neural network by a sufficiently large number of examples confirming its operation.

Deep learning

What is wrong: Stop or Speed Limit 45

From [9]:

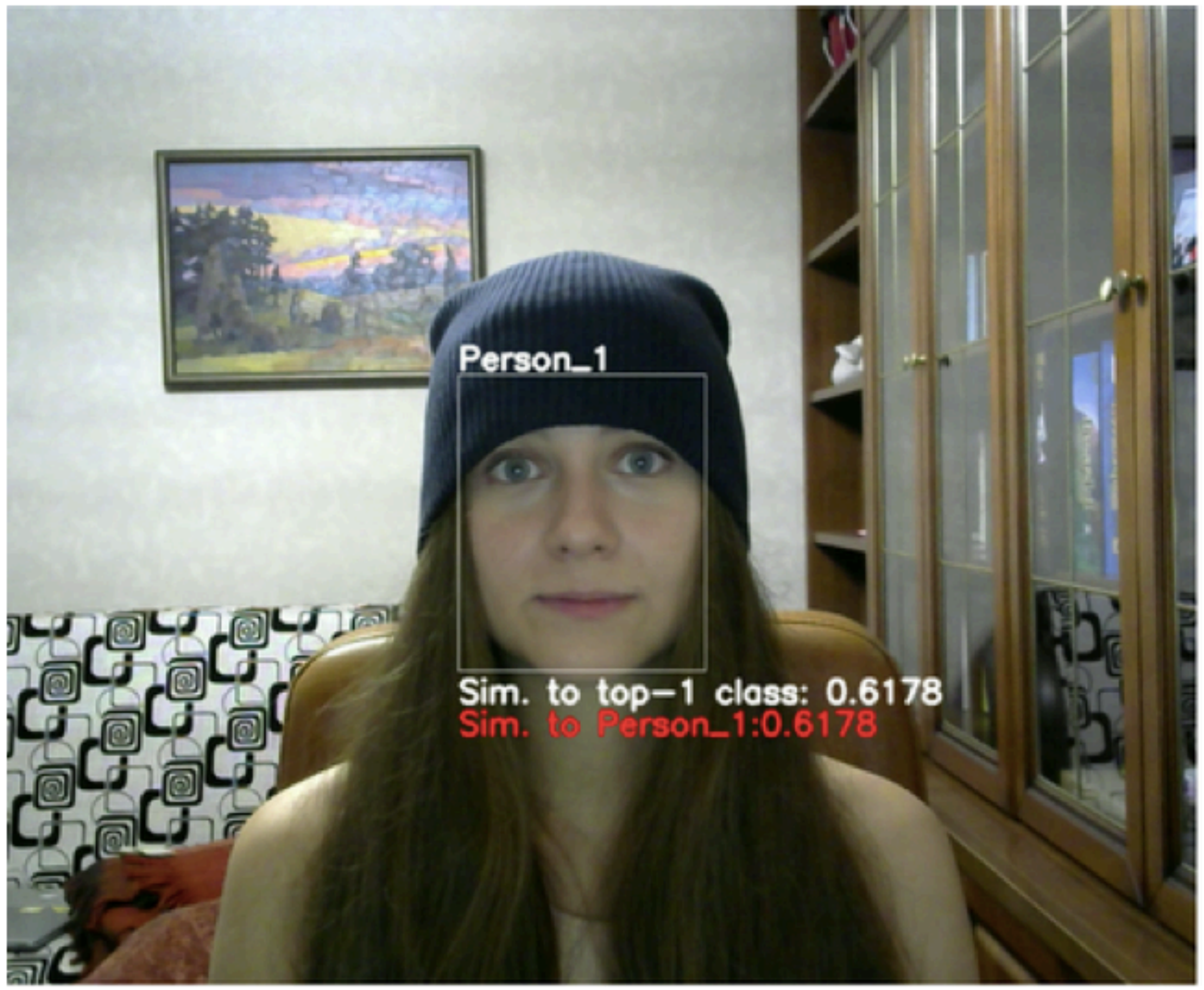


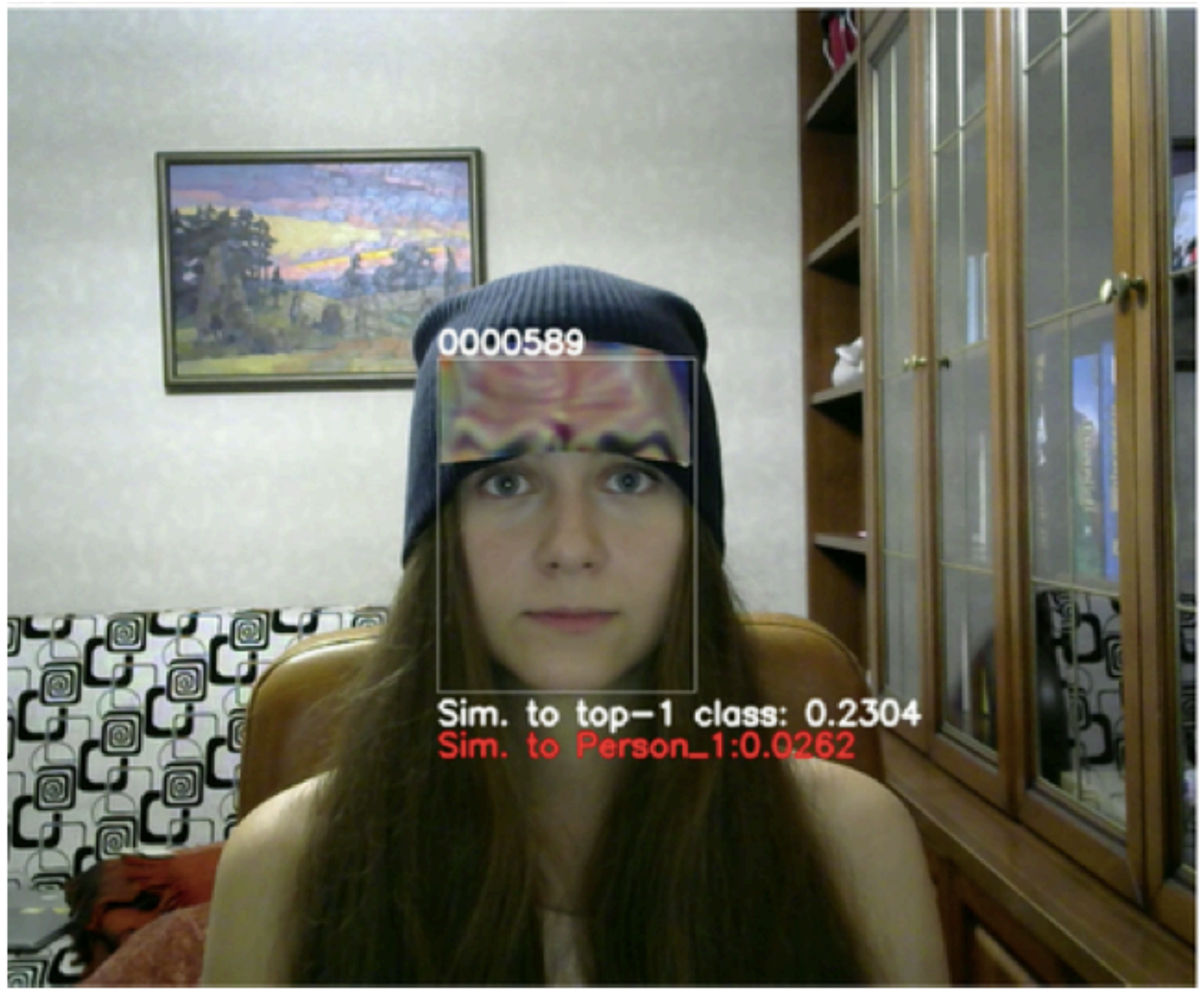
The left image shows real graffiti on a Stop sign, something that most humans would not think is suspicious. The right image shows our a physical perturbation applied to a Stop sign. We design our perturbations to mimic graffiti, and thus "hide in the human psyche."

Deep learning

What is wrong: Atack on Face ID system

From [10]:

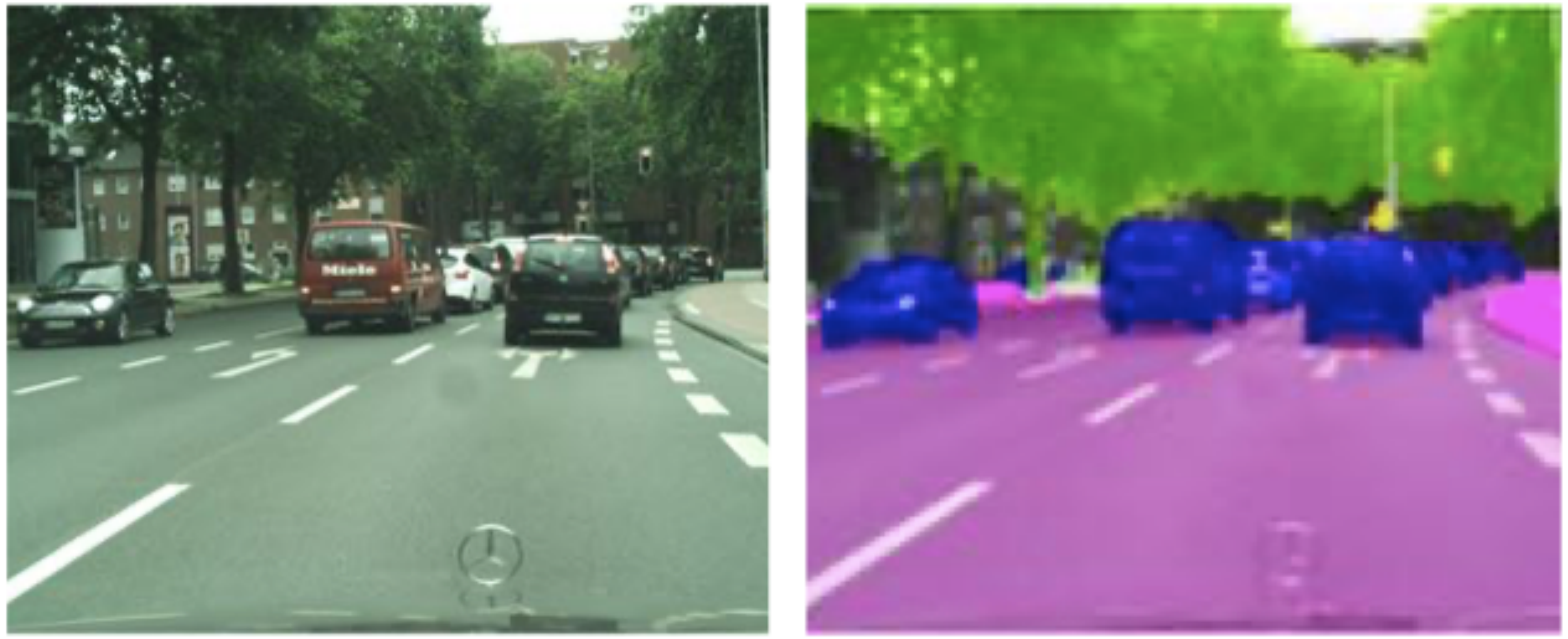




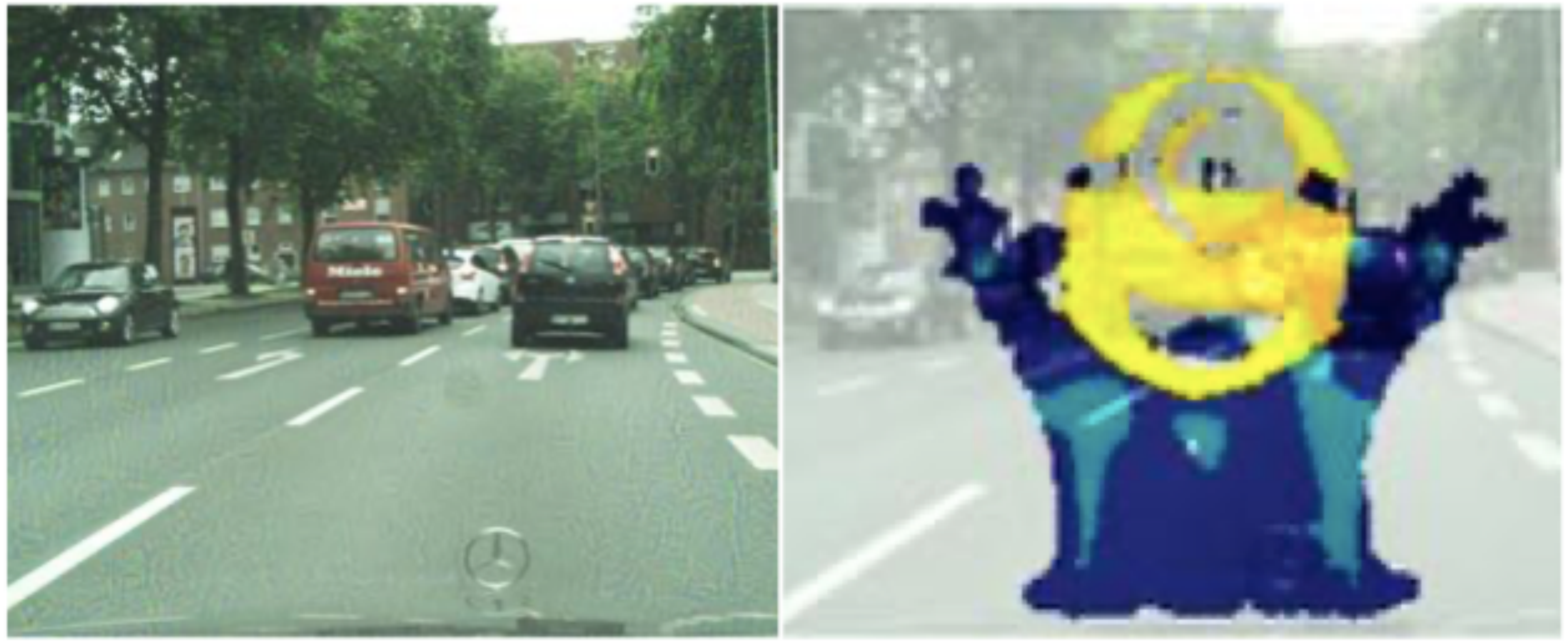
Deep learning

What is wrong: Image segmentation failure

From [11]:



Original semantic segmentation framework.



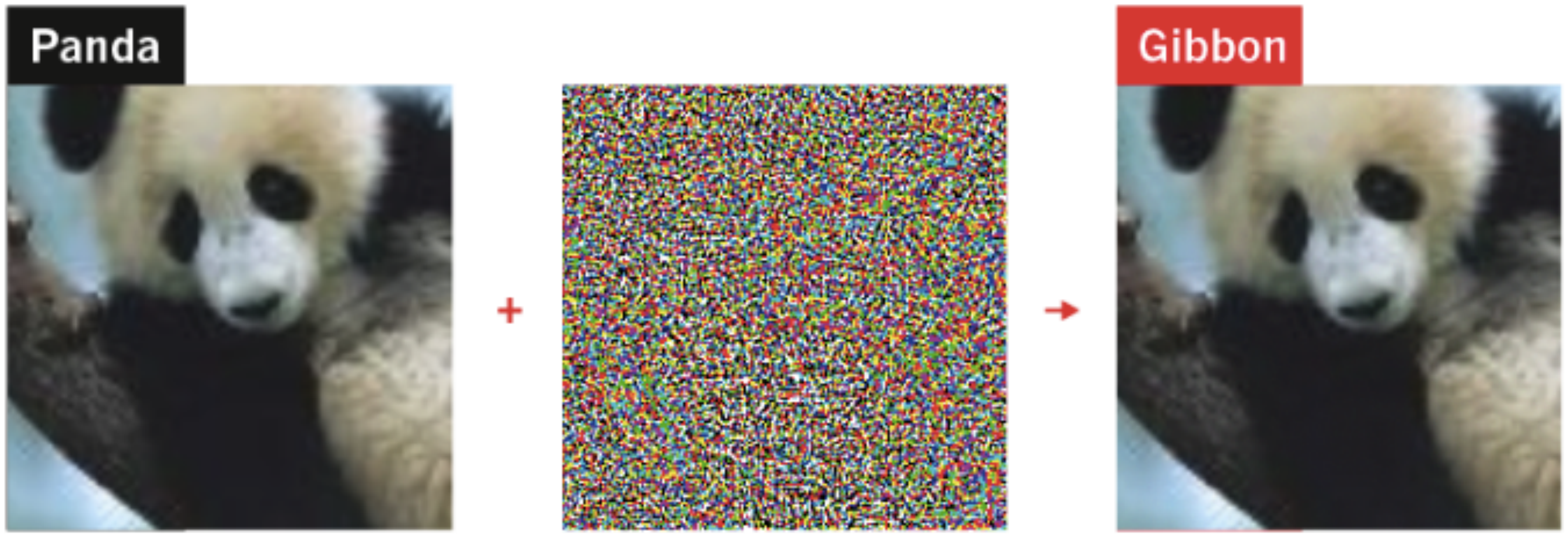
Compromised semantic segmentation framework.

Deep learning

What is wrong: Image segmentation failure

From [12]:

Adding carefully crafted noise to a picture can create a new image that people would see as identical, but which a DNN sees as utterly different.



Deep learning

What is wrong: Image segmentation failure

From [13]:

In their efforts to work out what's going wrong [with DNN], researchers have discovered a lot about why DNNs fail. "There are no fixes for the fundamental brittleness of deep neural networks," argues François Chollet, an AI engineer at Google in Mountain View, California.

Anybody who has played with machine learning knows these systems make stupid mistakes once in a while," says Yoshua Bengio at the University of Montreal in Canada, who is a pioneer of deep learning. "What was a surprise was the type of mistake," he says. "That was pretty striking. It's a type of mistake we would not have imagined would happen.

An AI trained to recognize aircraft might find that features such as patches of colour, texture or background are just as strong predictors as the things that we would consider salient, such as wings. But this also means that a very small change in the input can tip it over into what the AI considers an apparently different state [unlike a human for whom both states would represent the same information].

Bibliography

Bibliography

1. Mark S. Nixon, Alberto S. Aguado, *Feature Extraction and Image Processing for Computer Vision*, Academic Press (Elsevier), 2020 (4th edition)

2. Support-vector machine (Wikipedia),
https://en.wikipedia.org/wiki/Support-vector_machine
Retrieved 2021-12-28.

3. Drew Wilimitis, *The Kernel Trick in Support Vector Classification*,
https://towardsdatascience.com/the-kernel-trick-c98cdbcaeb3f
Retrieved 2021-12-30.

4. The Kernel Trick - THE MATH YOU SHOULD KNOW!
https://www.youtube.com/watch?v=wBVSbVktLIY
Retrieved 2021-12-30.

5. 16. Learning: Support Vector Machines
https://www.youtube.com/watch?v=_PwhiWxHK8o
Retrieved 2021-12-30.

6. Hornik K., Stinchcombe M., White H., *Multilayer feedforward networks are universal approximators*, Neural Networks, 2(5), 1989, s. 359-366.

7. xxx

8. Artificial neural network, https://en.wikipedia.org/wiki/Artificial_neural_network

9. Eykholt K., Evtimov I., Fernandes E., Li B., Rahmati A., Xiao Ch., Prakash A., Kohno T., Song D., *Robust Physical-World Attacks on Deep Learning Visual Classification*, 2018 https://arxiv.org/pdf/1707.08945.pdf

10. Komkov S., Aleksandr Petiushko, Advhat: *Real-World Adversarial Attack On Arcface Face Id System*, https://arxiv.org/pdf/1908.08705.pdf

11. Cisse M., Neverova N., Adi Y., Keshet J., *Houdini: Fooling Deep Structured Prediction Models*, 2017 https://arxiv.org/pdf/1707.05373.pdf

12. Douglas Heaven, Why deep-learning AIs are so easy to fool. Artificial-intelligence researchers are trying to fix the flaws of neural networks, 09 October 2019, Nature 574, 163-166 (2019), doi: https://doi.org/10.1038/d41586-019-03013-5, https://www.nature.com/articles/d41586-019-03013-5