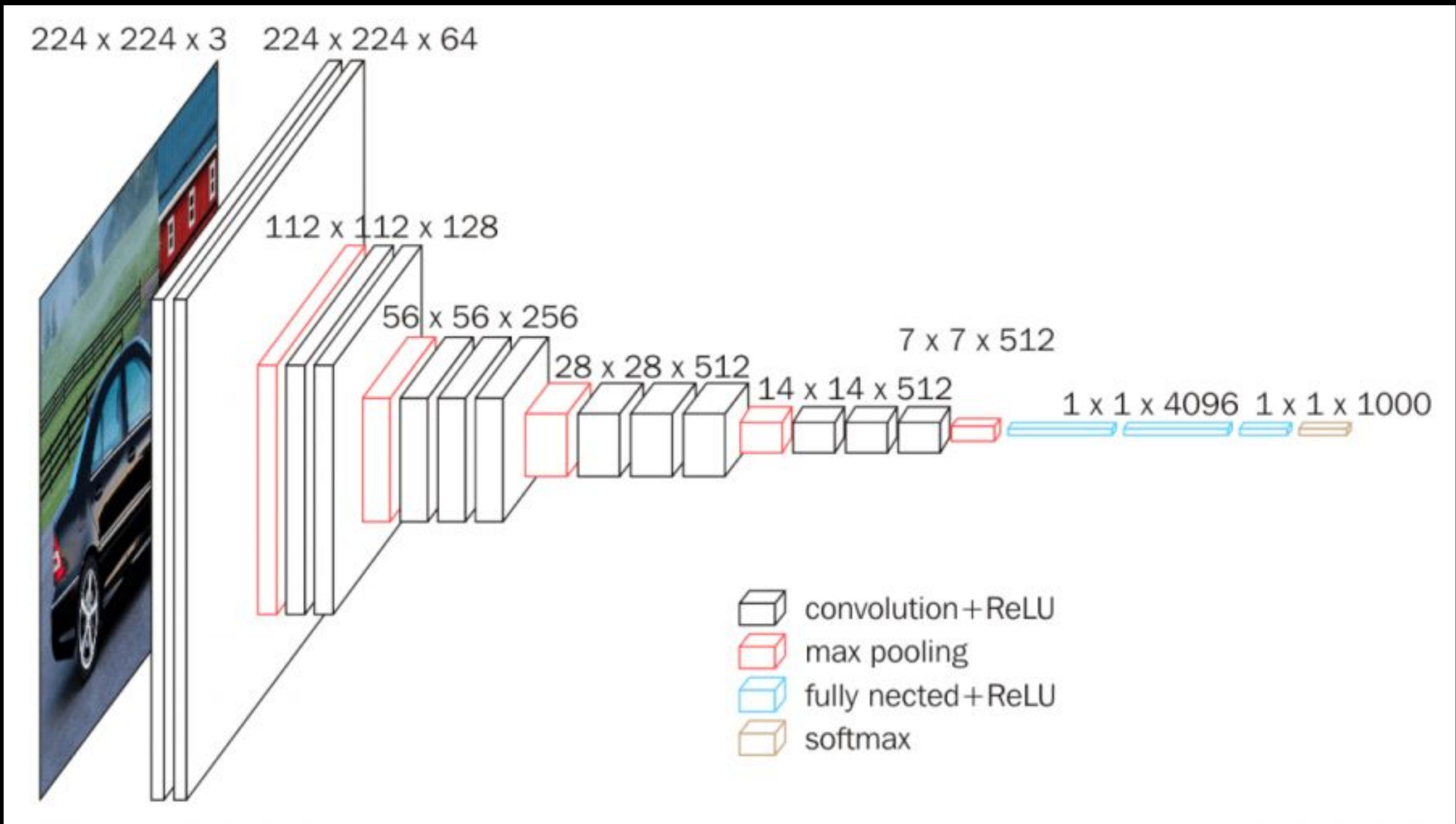
The big C in CNNs.



Convolutional Neural Networks have **outperformed humans** in image recognition tasks already in 2012.

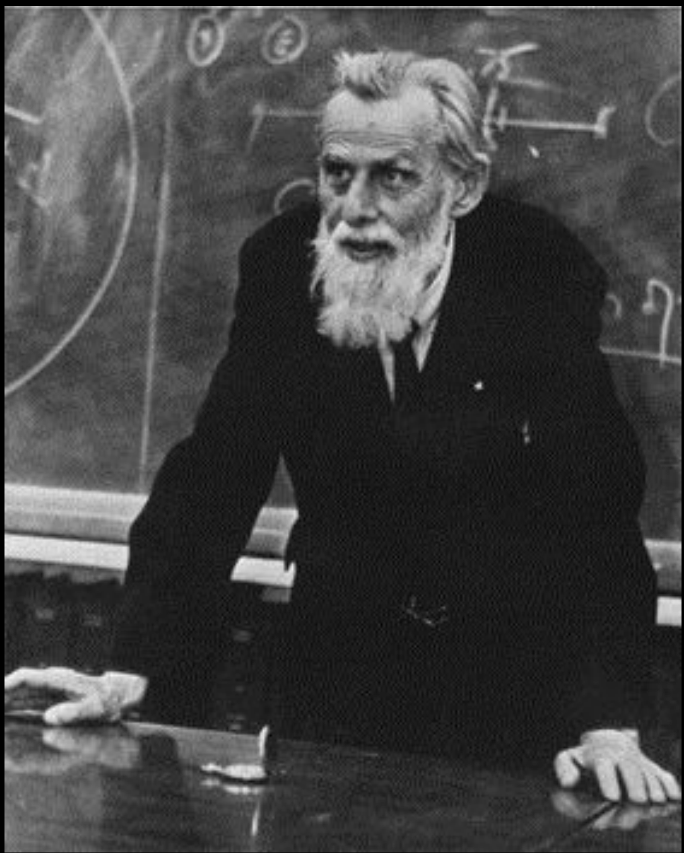
But what exactly is **convolution** and why does it matter?

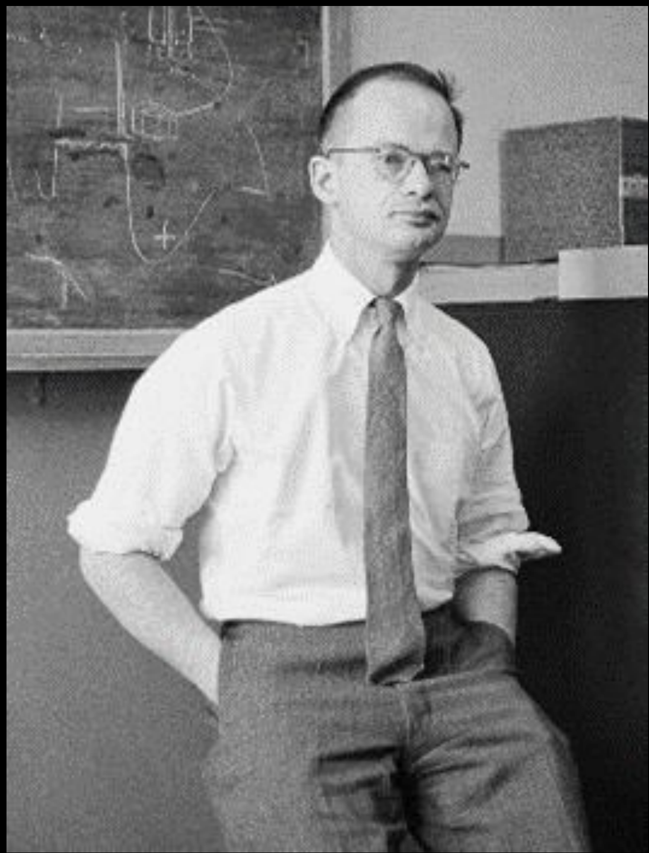
To answer this question we should look at the beginnings of Machine Learning.

The term machine learning was coined in 1959 by Arthur Samuel, a pioneer in AI and machine learning.

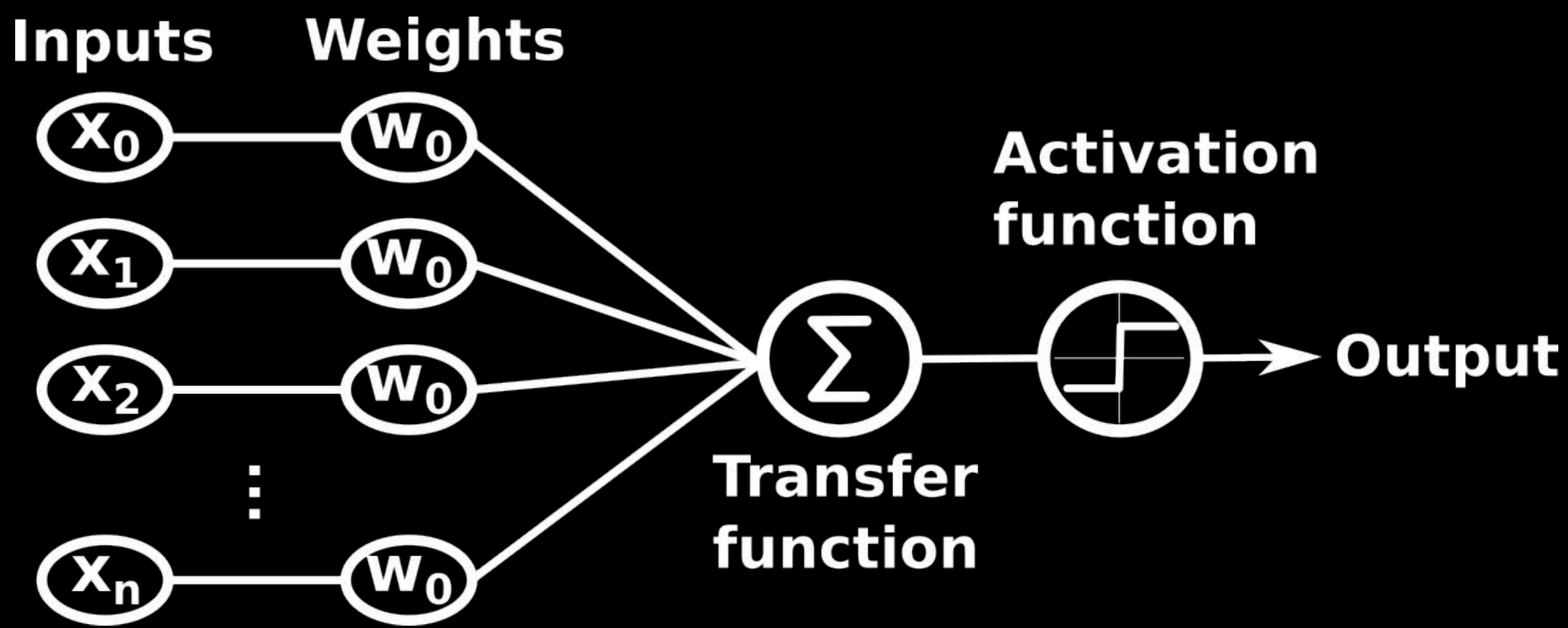
Arthur is known for his groundbreaking work in computer checkers, that optimized their strategy based on a search tree of board positions.

The first neuron model was already introduced in 1943 by McCulloch and Pitts.



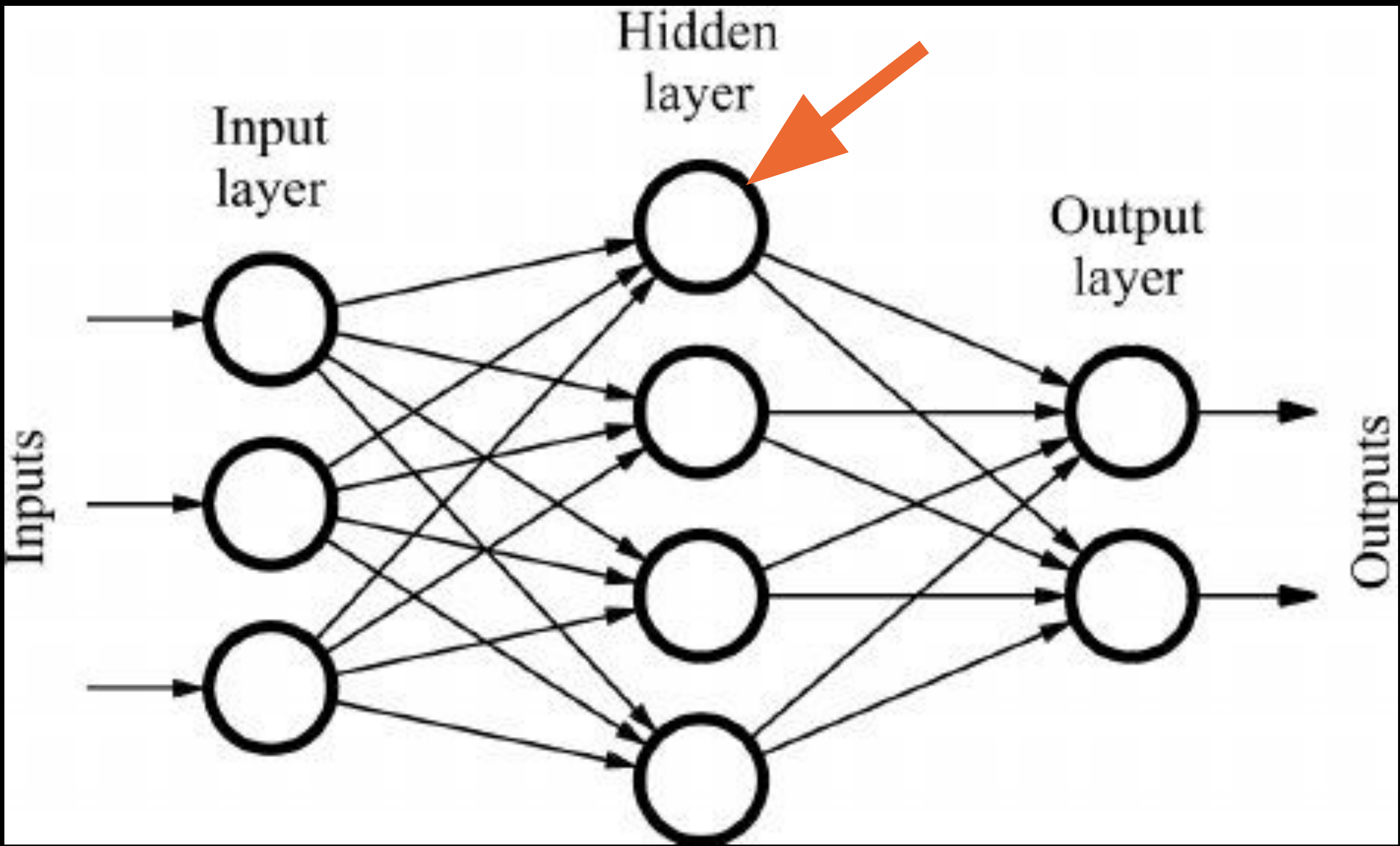


McCullochs and Pitts model was not able to learn, because it was based on a **hardcoded threshold** activation function.



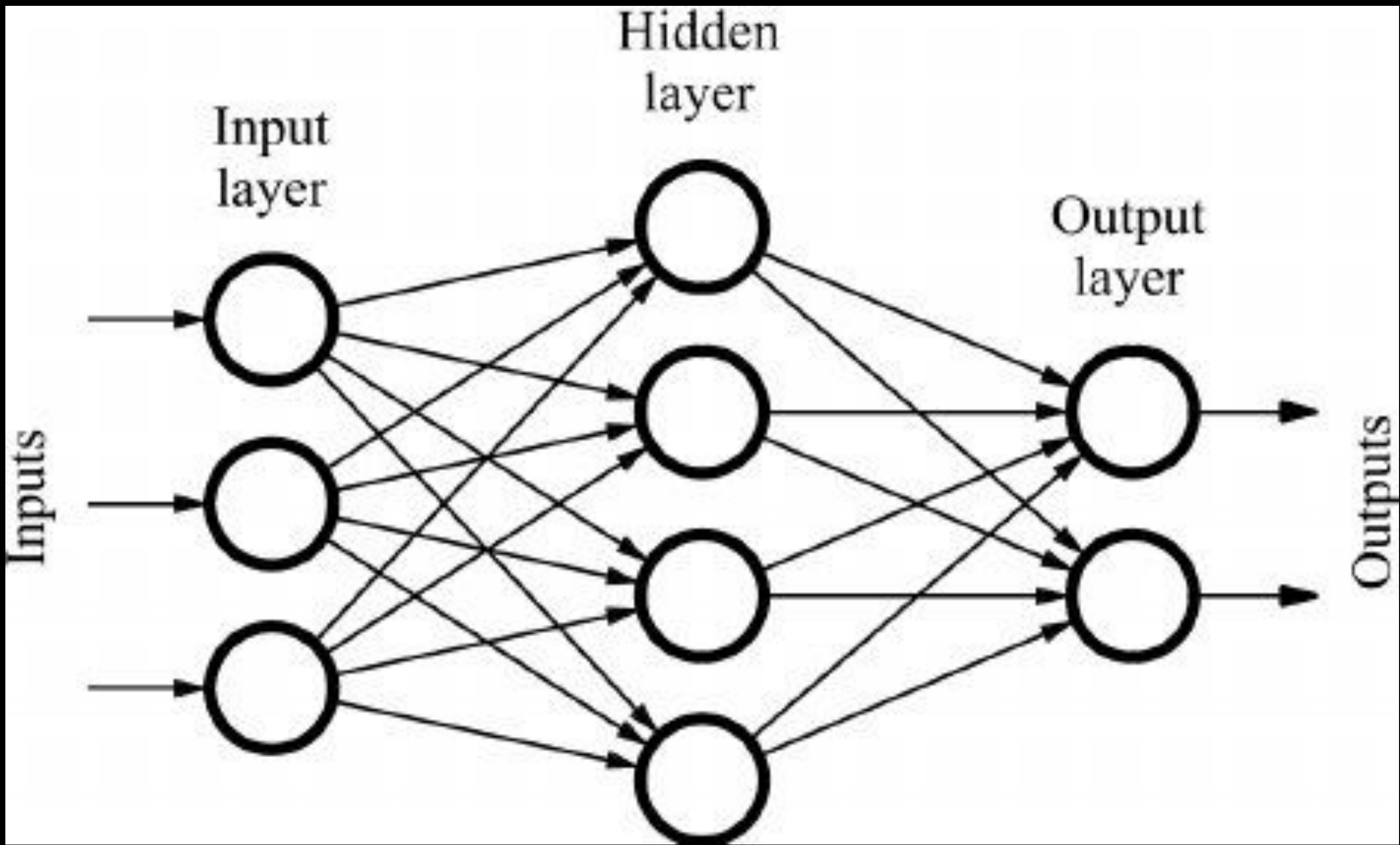
This changed with the first "Single layer Feed Forward Network" by Frank Rosenblatt in 1958.

In a feed forward network the nodes act as weights that are multiplied with the data going through.



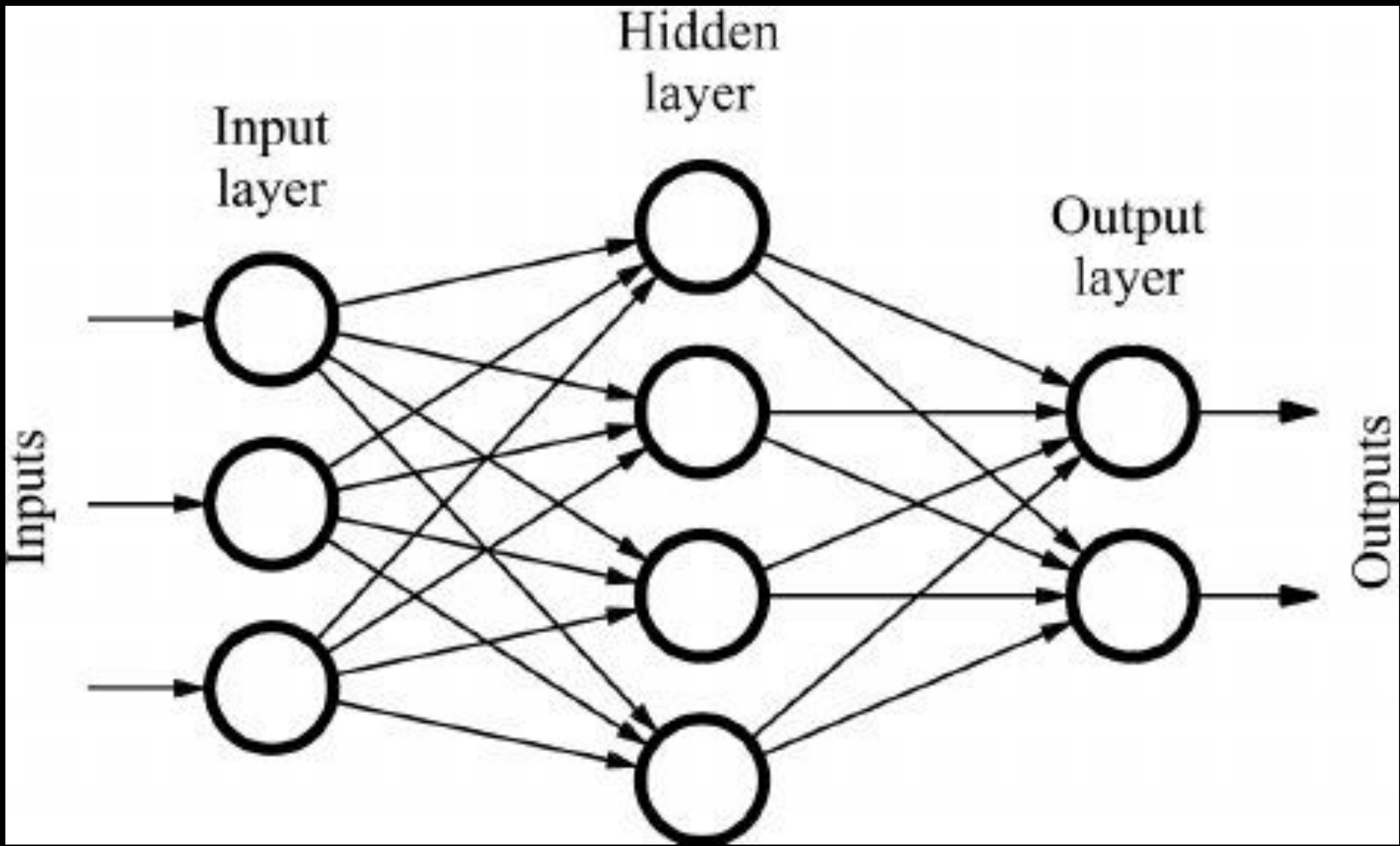
Source: Ramon Quiza

If the output is above a certain threshold, then the network outputs 1, otherwise 0.



Source: Ramon Quiza

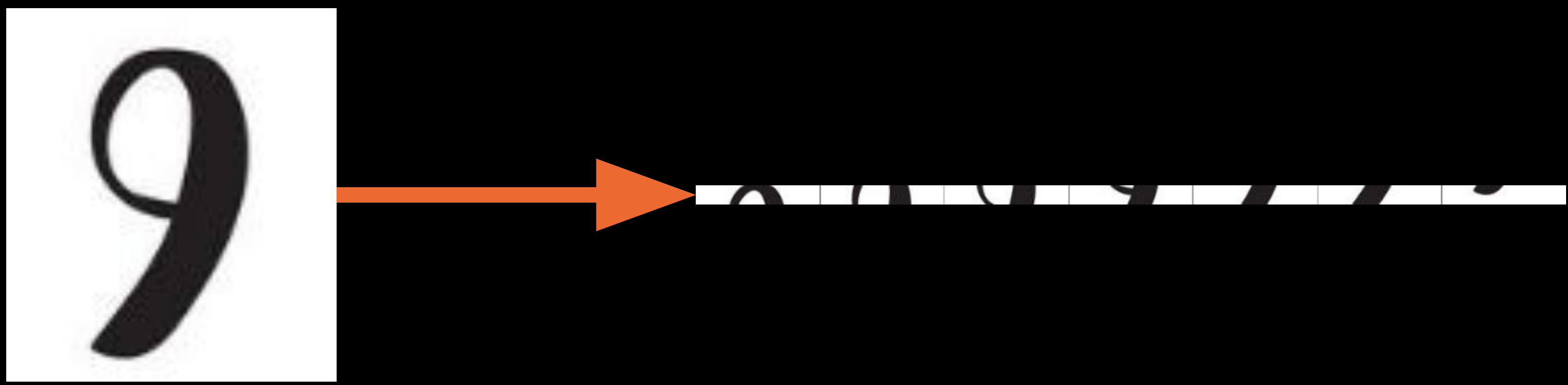
The network is trained, by adjusting the weights to better classify training data.



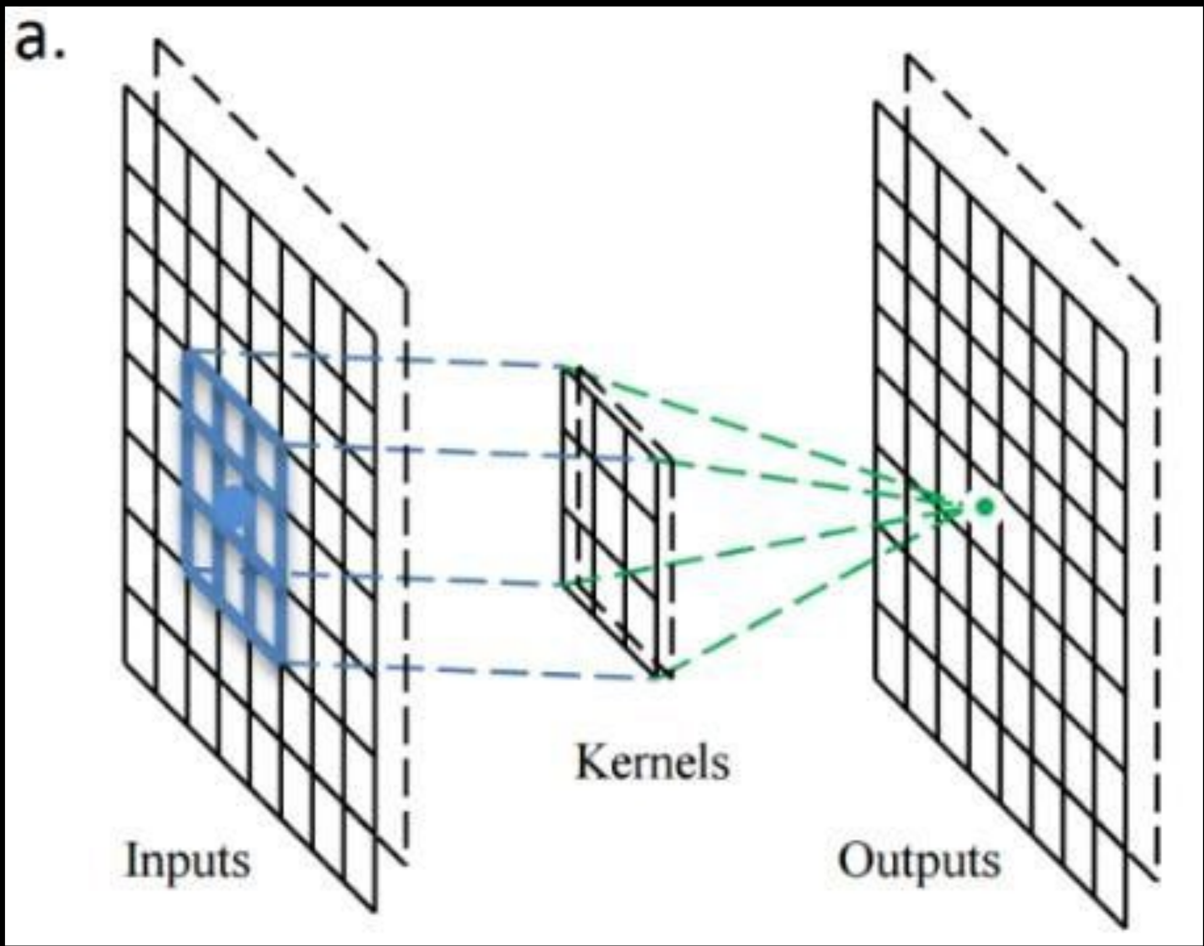
Source: Ramon Quiza

While the network architecture became more complex, these models had one fundamental drawback. The input was always one dimensional and every input neuron interacted with every output neuron.

An image for example would need to be flattened and all of it’s pixels would be processed the same, losing valuable neighborhood information.



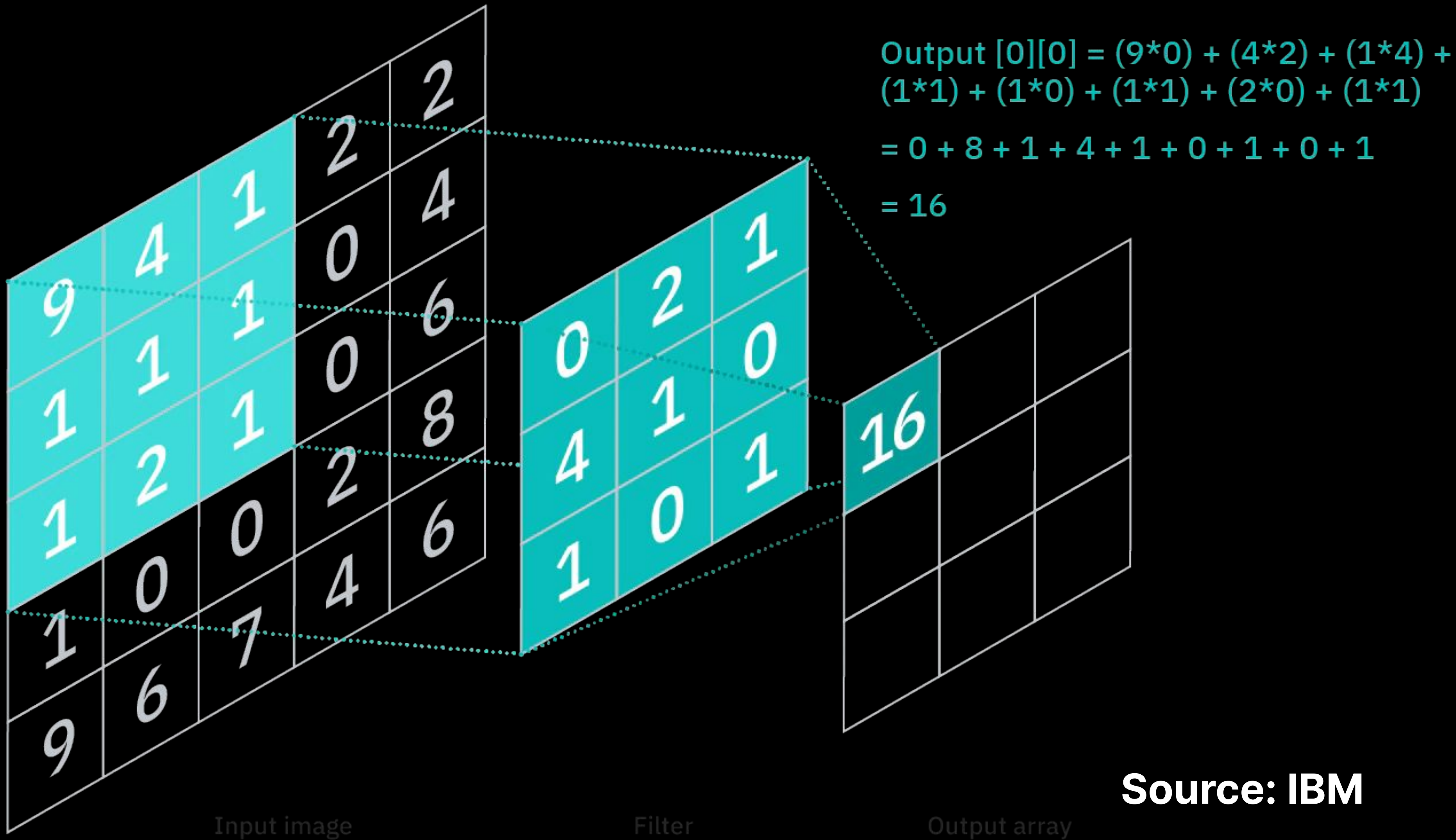
Convolutional neural networks or CNNs, solve this, by using a convolutional kernel.



Source: Konstantin S. Pervunin

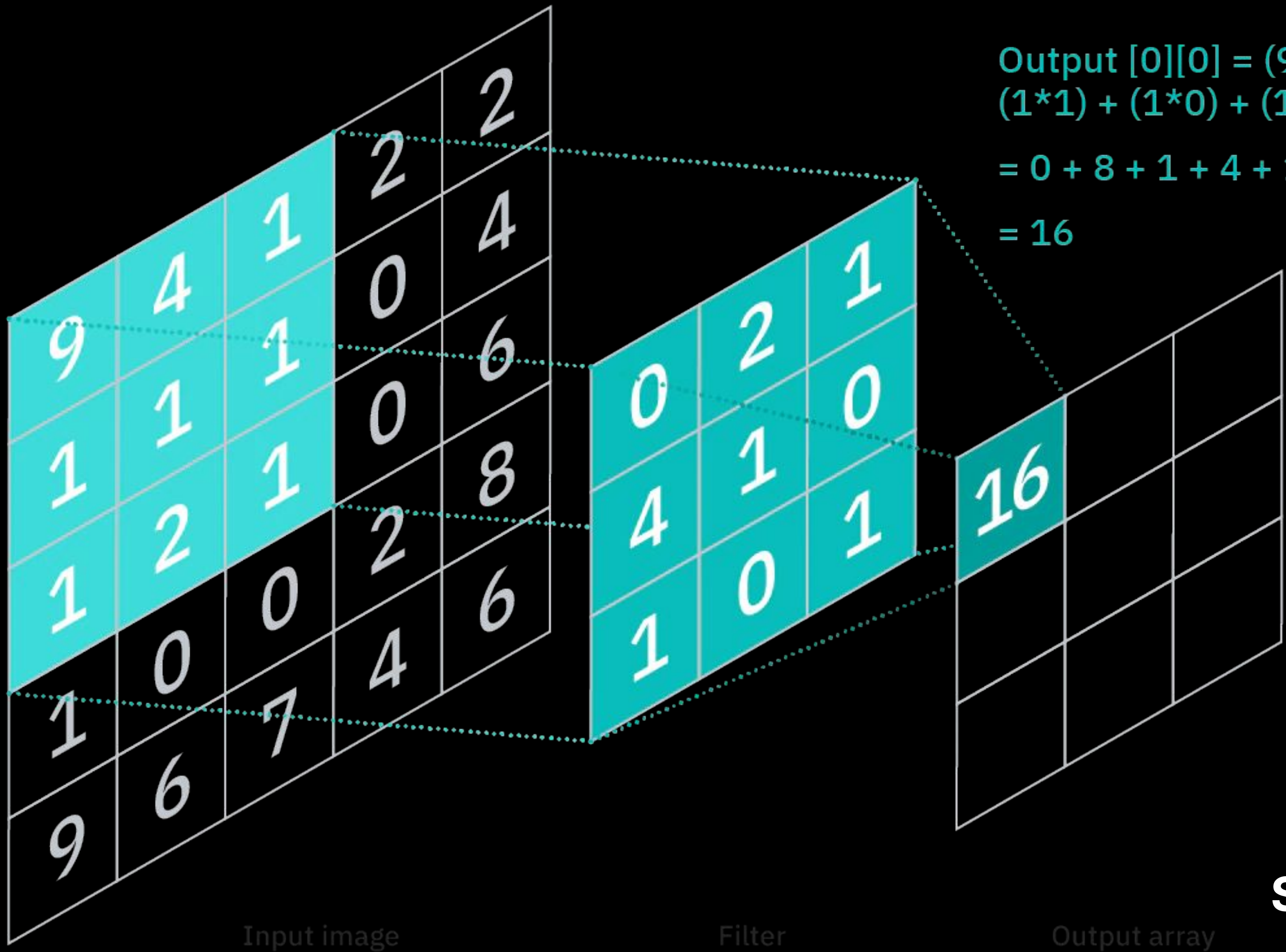
CNNs were introduced in 1989 by Yann LeCun et al. in a revolutionary paper on handwritten digit recognition.

Convolutional kernels are **small filters** that look at a small part of the input image.



Source: IBM

Each pixel is multiplied by the respective value in the kernel.



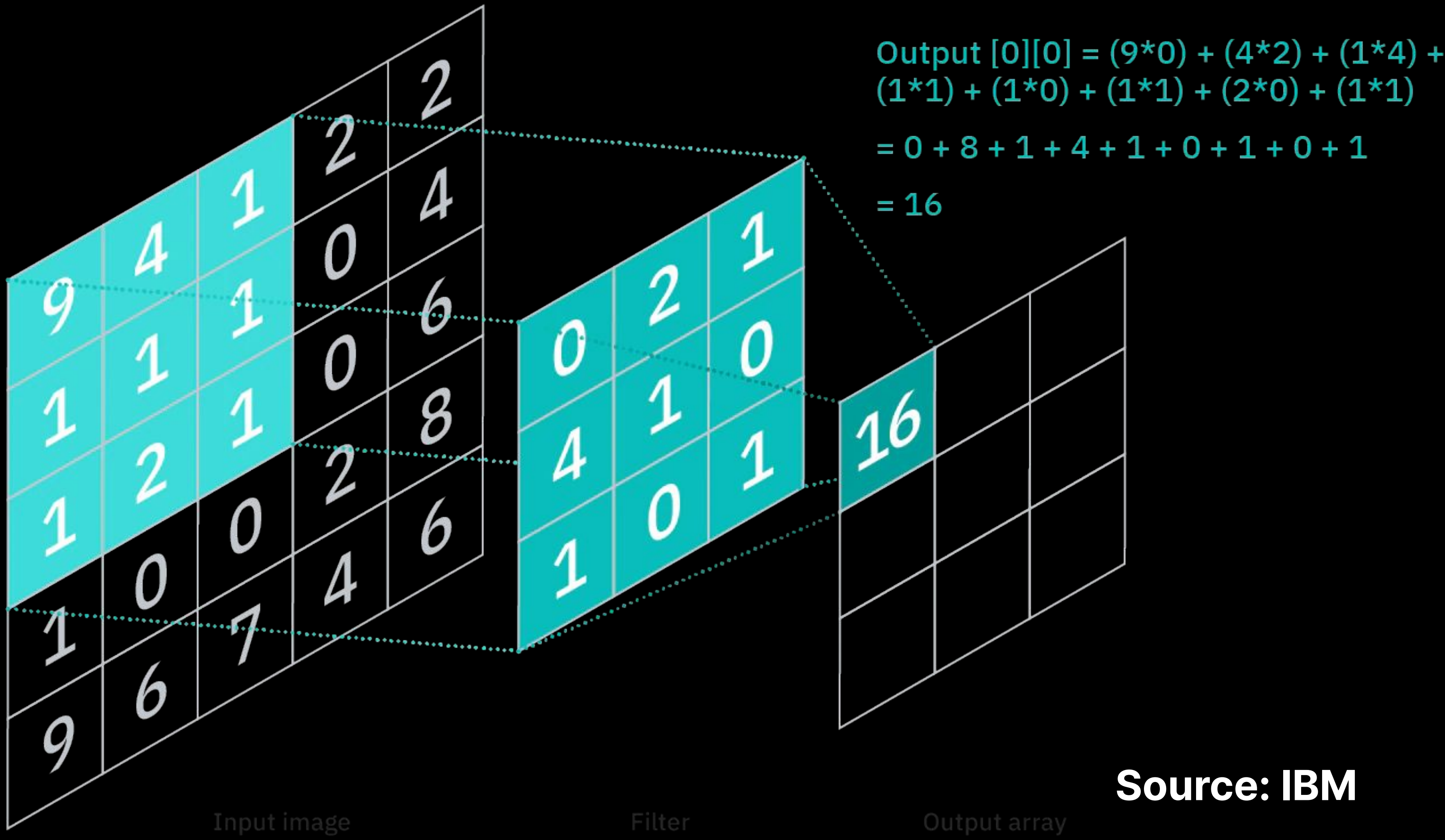
Output [0][0] = (9*0) + (4*2) + (1*4) + (1*1) + (1*0) + (1*1) + (2*0) + (1*1)

= 0 + 8 + 1 + 4 + 1 + 0 + 1 + 0 + 1

= 16

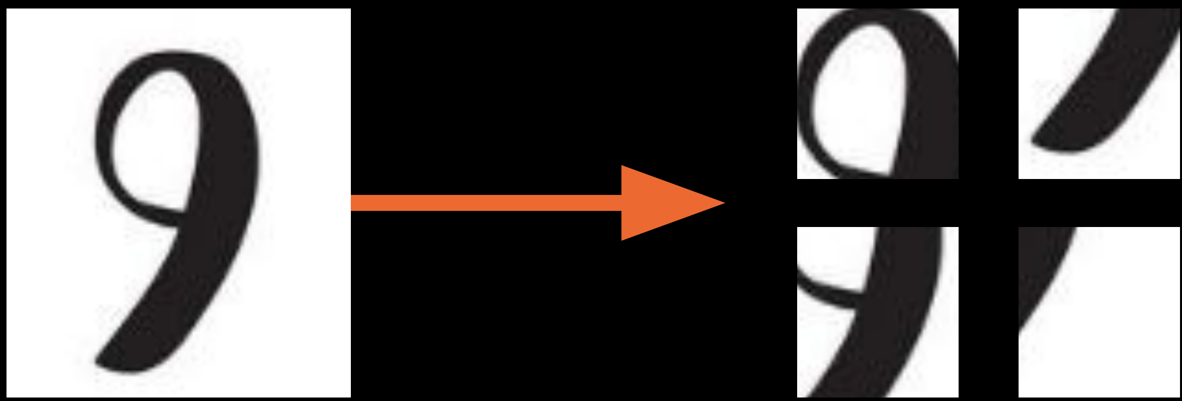
Source: IBM

The result is **summed up** for a single value and stored in a smaller grid.



Source: IBM

You can think of a kernel as a **small window** that only looks at a small part of the image. Instead of processing all the pixels we only look at a small subset.



This provides several advantages at once. By looking at a subset of the image in each kernel, the network has a sparse interaction between the neurons. Improving the efficiency and size of the model.

Assuming that if a feature is useful to compute in one position it should also be useful to compute elsewhere, CNNs have one filter kernel for each convolution step. This is called parameter sharing.

Parameter sharing does not only drastically reduce the model complexity but it also guarantees **equivariant representation** for the input data. If the input is changed in a way the output will change in the same way.

Now you should know that the convolutional kernel was a game changer in deep learning, providing several advantages with one idea.

Remember

1. CNN successfully recognized handwritten digits already in 1989.

2. CNNS have sparse interaction due to a local convolution filter.

3. CNNs utilize parameter sharing by using a constant kernel in each convolution layer.

Feel free to reach out or to connect with me for more weekly slideshows on visualization, data science and machine learning.