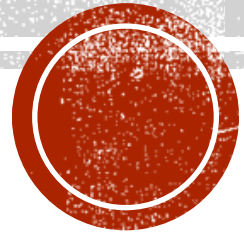
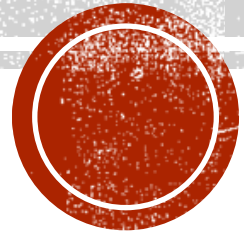
CONVOLUTIONAL NEURAL NETWORKS (CNN)

Dr. Brian Mc Ginley



CNN BACKGROUND

Dr. Brian Mc Ginley



EXAMPLE

- Imagine we wanted to read the time from these images:
- We can see how the raw pixels would require a complex model to determine the time.
- Simple edge detection algorithms could calculate the better features perfectly.
- From there we could represent these directly into polar form to give us angles and lengths so we can determine the angle of the hour hand and minute hand.
- From there we could have a mathematical equation or a look up table.

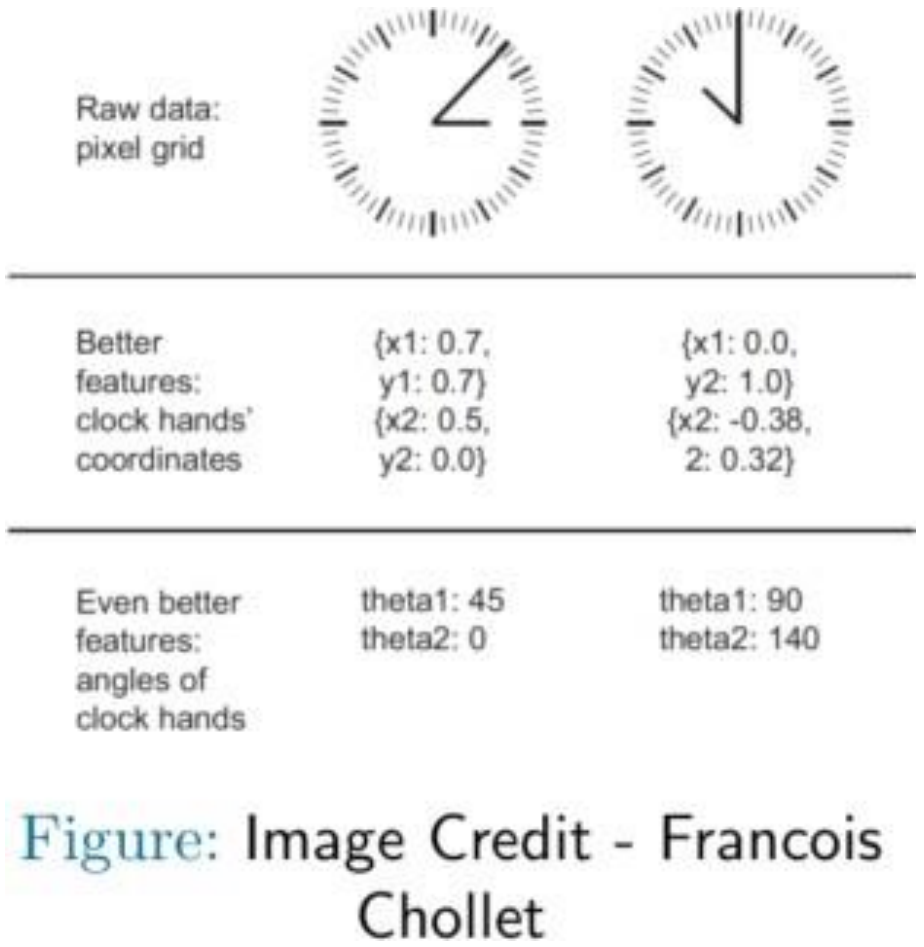


Figure: Image Credit - Francois Chollet

CONVENTIONAL APPROACH

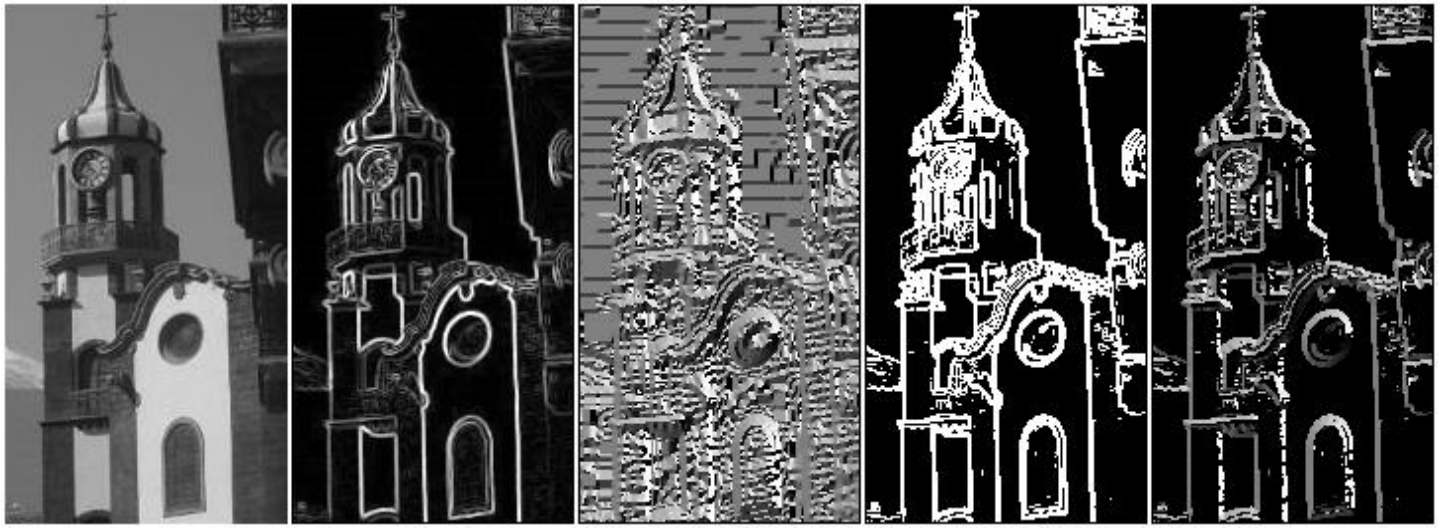
-1	0	1
-2	0	2
-1	0	1

G_x

-1	-2	-1
0	0	0
1	2	1

G_y

- Hand-crafted algorithms such as Sobel and Canny were designed to detect edges, and algorithms such as Harris, Shi-Tomasi and SIFT were created to detect corners

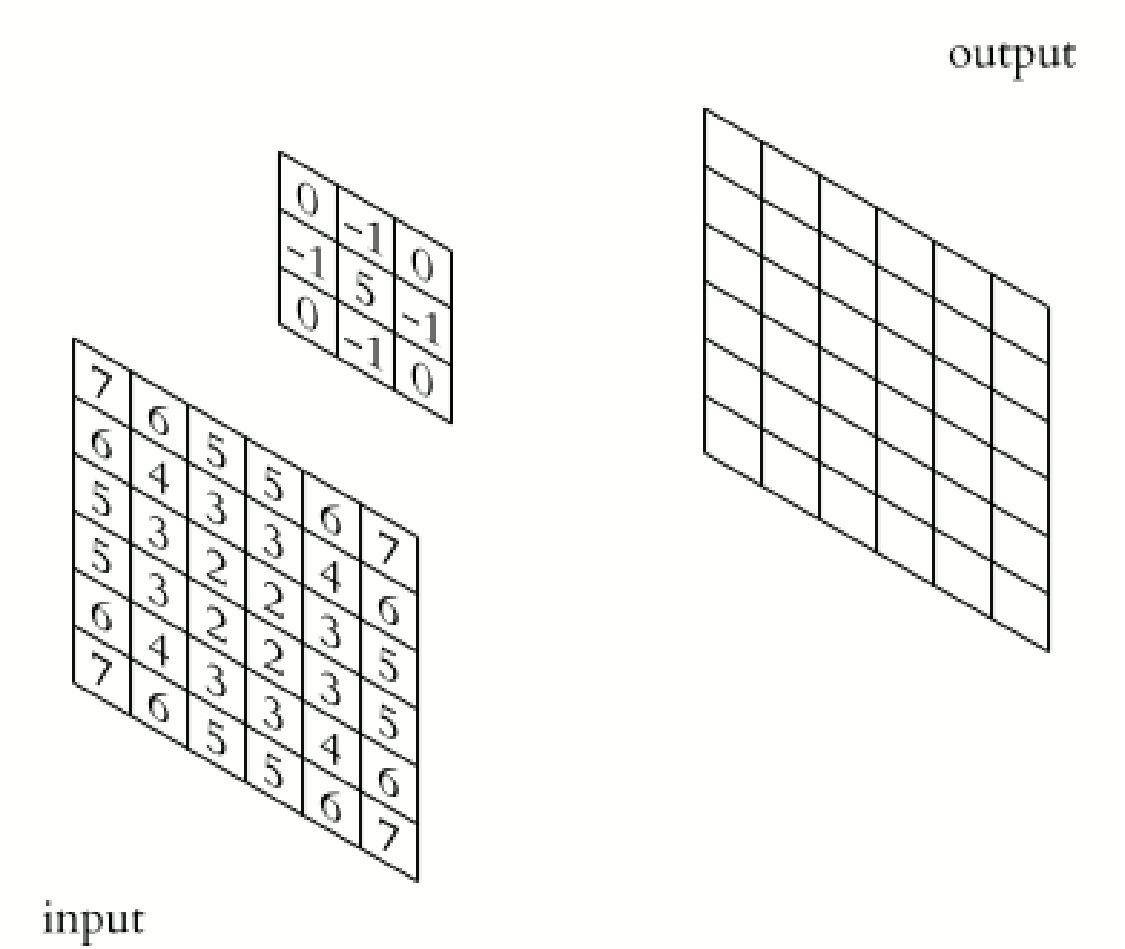


DOT PRODUCT & CONVOLUTION

$$\mathbf{a} = \langle a_1, a_2, a_3 \rangle \quad \mathbf{b} = \langle b_1, b_2, b_3 \rangle$$

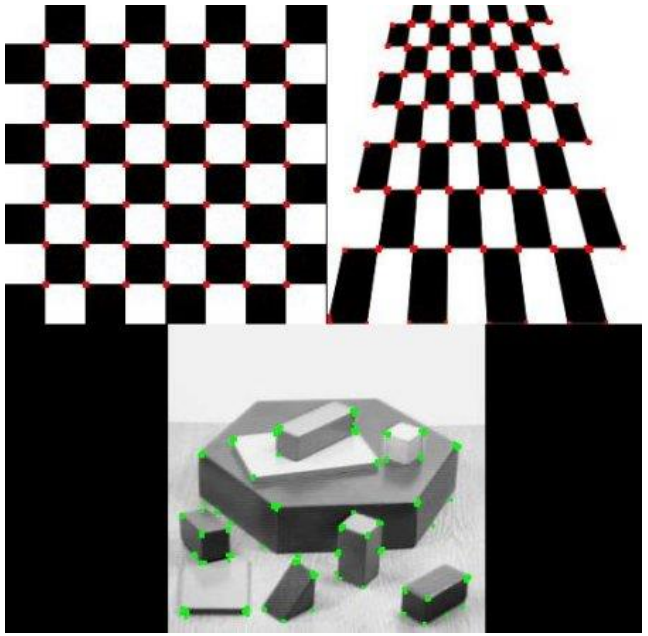
$$\boxed{\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2 + a_3 b_3}$$

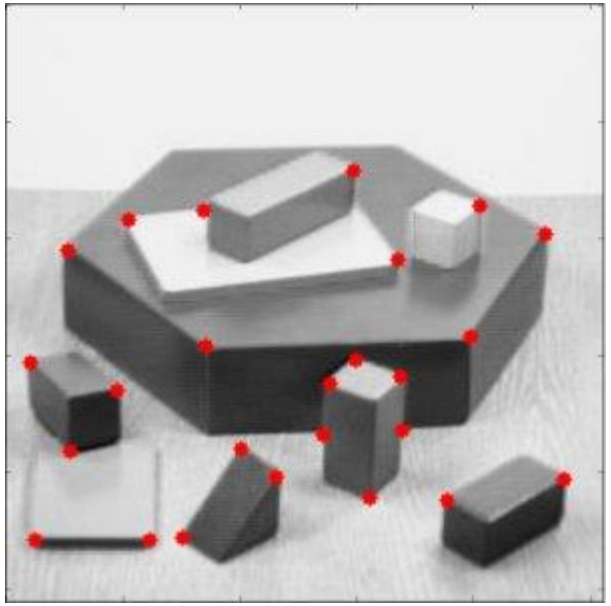
CONVOLUTION AND DOT PRODUCT



CONVENTIONAL APPROACH

- Hand-crafted algorithms such as Sobel and Canny were designed to detect edges, and algorithms such as Harris, Shi-Tomasi and SIFT were created to detect corners





REPRESENTATION LEARNING

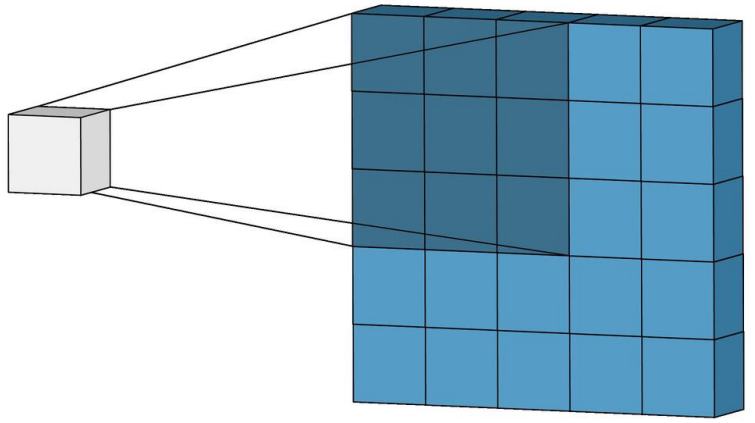
- We could try getting our machine learning algorithm to discover not just the mapping from the original representation to the output, but also learn to create appropriate intermediate representations itself.
- The problem is that often it is difficult to make this representation change in just one jump.
- This is where Deep Learning comes in.
- The Deep in Deep Learning refers to the number of levels. Each level makes a change to the representation.
  - Until eventually we can separate classes with our favourite machine learning classifier.

LEARNED REPRESENTATIONS/FEATURES

- These learned representations can often result in better performance than hand engineered representations.
- Often, the lower-level features are universally useful for similar tasks.
- This means that we can adapt to new tasks with relative ease, by only retraining part of the network; A process called fine tuning or Transfer Learning.
- Learning representations takes longer and requires a much more complex model than when we supply the hand-engineered features.
- The process of hand engineering features for a complex task requires a lot of human ingenuity and time; Think decades, for a community of researchers.

THE CURSE OF DIMENSIONALITY

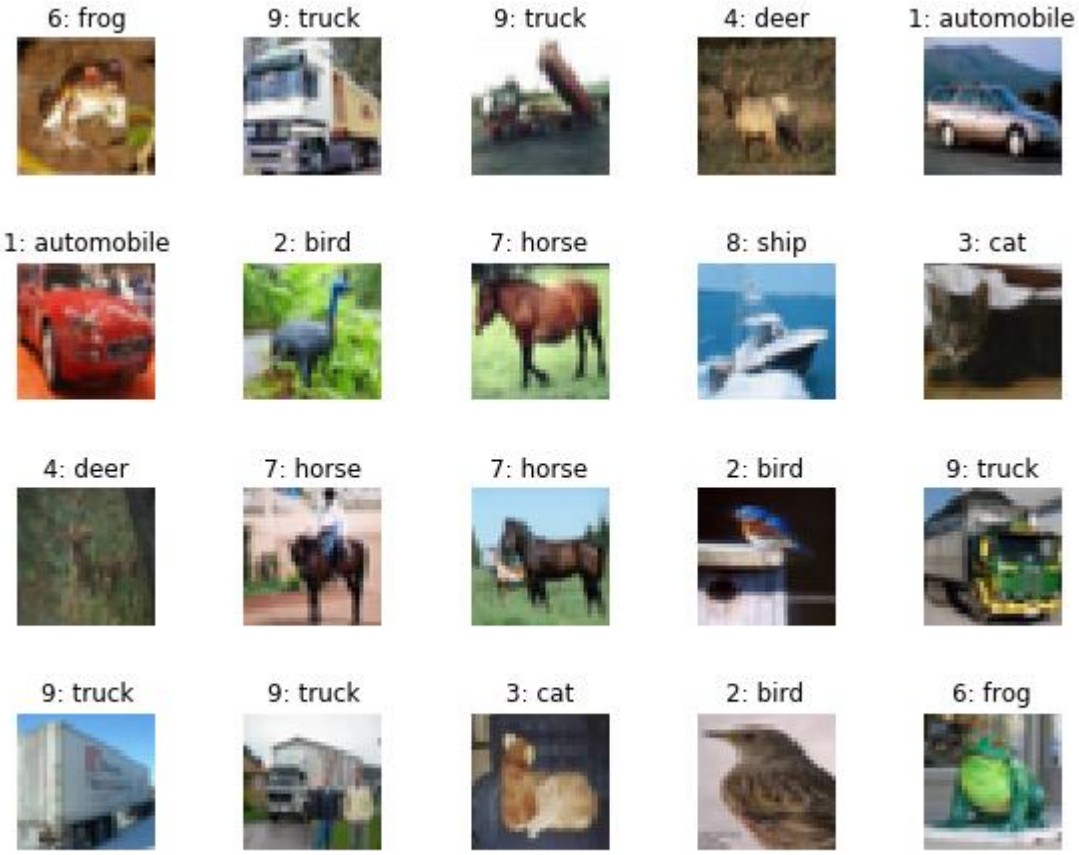
- Images are big.
- Really big.
- They make other types of problems look like toy problems.
- The difficulty is that in a single image there is a huge amount of data, and this makes image problems very high dimensional.
- To have a hope of using images in machine learning you have to be clever and think about the special nature of images.



From: Intuitively Understanding Convolutions for Deep Learning | by Irhum Shafkat | Towards Data Science

HOW MANY DIMENSIONS

- Images are, on the face of it two dimensional.
- If they are colour, then they are three dimensional as you have three channels (usually RGB but other colour spaces are available).
- Three dimensions doesn't seem that high.
- If we consider an image that's 32 × 32 × 3 then from a machine learning point of view this is actually the same as 3072 × 1, i.e. a 3072 dimensional problem.
- And that's for a tiny image.



UTILISING 2D

- We know that the image has a 2D structure.
- If you take all the pixels of an image and mix them up and put them together in a different order, you do not have the same thing.
- i.e. the features/dimensions are not independent of their position. So, if we can in some way modify our algorithm to recognise this then we may have a means of tackling enormous problems.
- The key insight is spatial invariance.
- What this says is, if we are trying to find something in an image, it doesn't matter where in the image it is, if it's there it's there.
- And if we place a filter over it, we should get the same response. The idea that cracked this was the Convolutional Neural Network (CNN).

HISTORY

- 1989 – (published in 1998) LeNet by Yann leCun was the first successful use of a CNN. It was successfully deployed for use in postal services for reading handwritten postal codes (Convolutional Network Demo from 1989).
- 2012 – AlexNet was the breakthrough for CNNs. Alex Krishevsky, Ilya Sutskever and Geoffrey Hinton created a network that won the ImageNet challenge by a huge margin.
- 2014 - GoogLeNet/Inception - winner of the ImageNet 2014 competition was GoogLeNet - reduced the number of parameters from 60 million (AlexNet) to 4 million. Took a trained human to beat it
- 2014 – VGGNet (runner-up at the ImageNet 2014). Most preferred choice for extracting features from images. The trained VGGNet weights are publicly available and are used in many other applications as a baseline feature extractor.
- 2015 – ResNet - winner at ImageNet 2015. Entry from Microsoft. Beats human level performance on the dataset.

RESOURCES

- Francois Chollet, the creator of Keras, has a great thread to get you started in using Keras with TensorFlow2: https://threadreaderapp.com/thread/1105139360226140160.html
- Play around with the Deep Dream Generator: https://deepdreamgenerator.com/
- CS231n Convolutional Neural Networks for Visual Recognition from Stanford can be useful notes too: CS231n Convolutional Neural Networks for Visual Recognition
- Ian Goodfellow, Yoshua Bengio, Aaron Courville - Deep Learning Book
- Francois Chollet - Deep Learning with Python

LAYERS TO BUILD CONVNETS/CNNS

- A ConvNet is made up of Layers. Every Layer has a simple API: It transforms an input 3D volume to an output 3D volume with some differentiable function that may or may not have parameters.
- Three main types of layers to build CNNs:
  1. Convolutional Layer
  2. Pooling Layer
  3. Fully-Connected Layer