CV for Image Generative AI

AI image generators utilize trained artificial neural networks to create images from scratch. These generators have the capacity to create original, realistic visuals. What makes them particularly remarkable is their ability to fuse styles, concepts, and attributes to fabricate artistic and contextually relevant imagery. This is made possible through Generative AI, a subset of artificial intelligence focused on content creation.

AI image generators are trained on an extensive amount of data, which comprises large datasets of images. Through the training process, the algorithms learn different aspects and characteristics of the images within the datasets. As a result, they become capable of generating new images that bear similarities in style and content to those found in the training data.

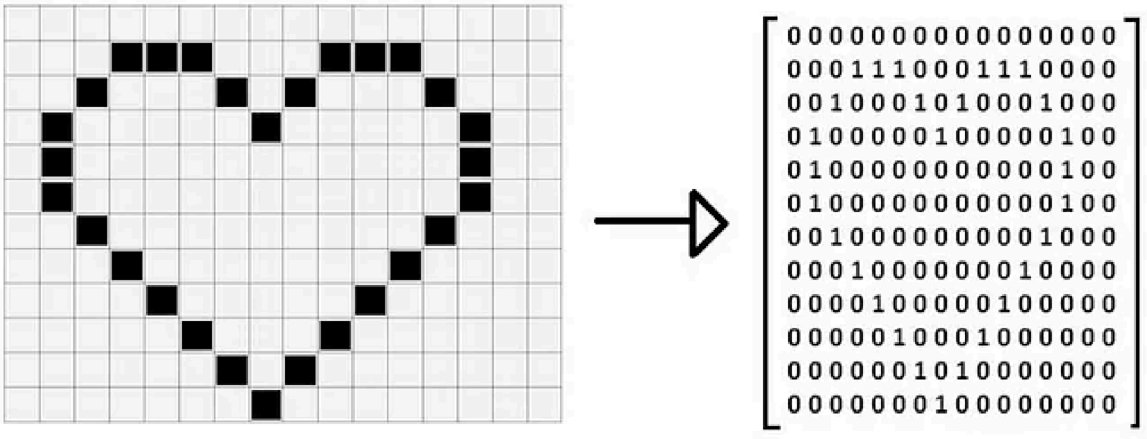
Images as seen by Computers

Before we move forward on how to generate images lets briefly understand how a computer views and understands images and videos.
An image is represented as a grid of pixels (**an image is a 2D projection of a 3D scene**). It is a continuous function of two coordinates in the image plane; usually expressed as (i,j) or (column/width, row/height) or somewhat confusingly (y,x).

Binary Images: Let's start with the simplest type of image - a 2D binary image as shown below. Imagine a grid of squares, where each square is either black or white. This is essentially how a computer sees a binary image.

- Each square in this grid is called a pixel (short for "picture element").
- In a binary image, each pixel has only two possible values: 0 (black) or 1 (white).
- The computer stores this as a 2D array of 0s and 1s.

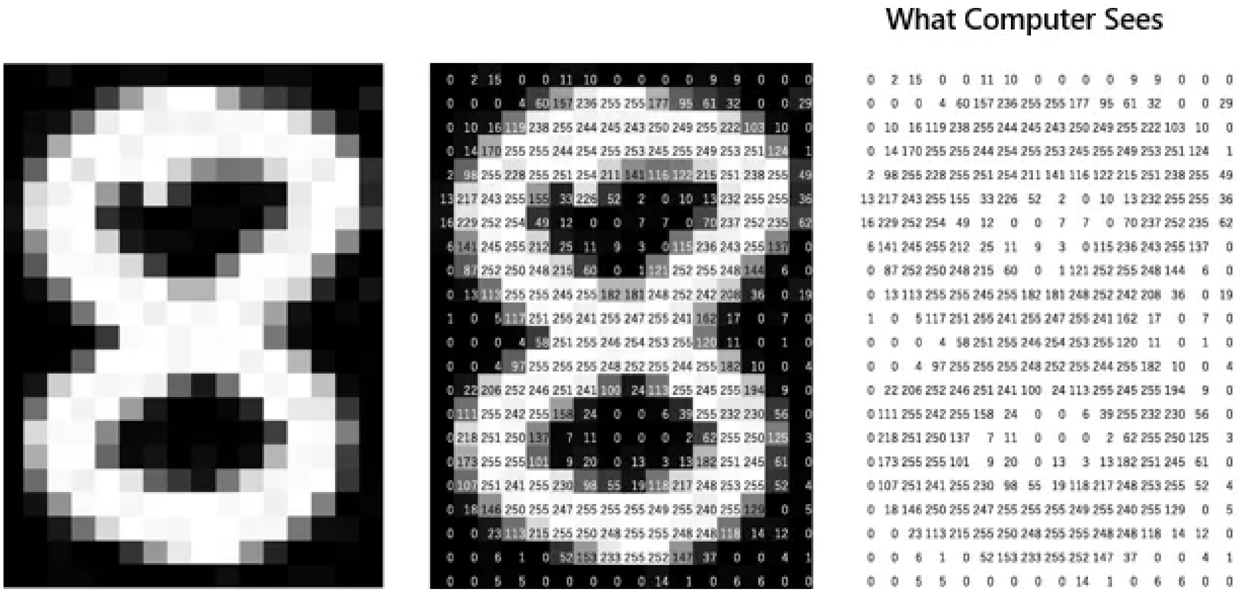
Say an image has a resolution of 100 x 200, this would imply our image is represented as a grid of pixels, with 100 rows and 200 columns, with the width being represented as the number of columns and the number of rows representing the height of the image. And overall, there are 100 x 200 = 30,000 pixels in our image



```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 1 1 0 0 0 1 1 1 0 0 0 0
0 0 1 0 0 0 1 0 1 0 0 0 1 0 0 0
0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0
0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0
0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0
0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0
0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
```

Grayscale Images: The next step up is grayscale images. Instead of just black and white, we now have shades of gray.

- Each pixel can have a value from 0 (black) to 255 (white), with numbers in between representing different shades of gray.
- The computer stores this as a 2D array of numbers from 0 to 255.
- The values between 0 and 255 are varying shades of gray, where values closer to 0 are darker and values closer to 255 are lighter.



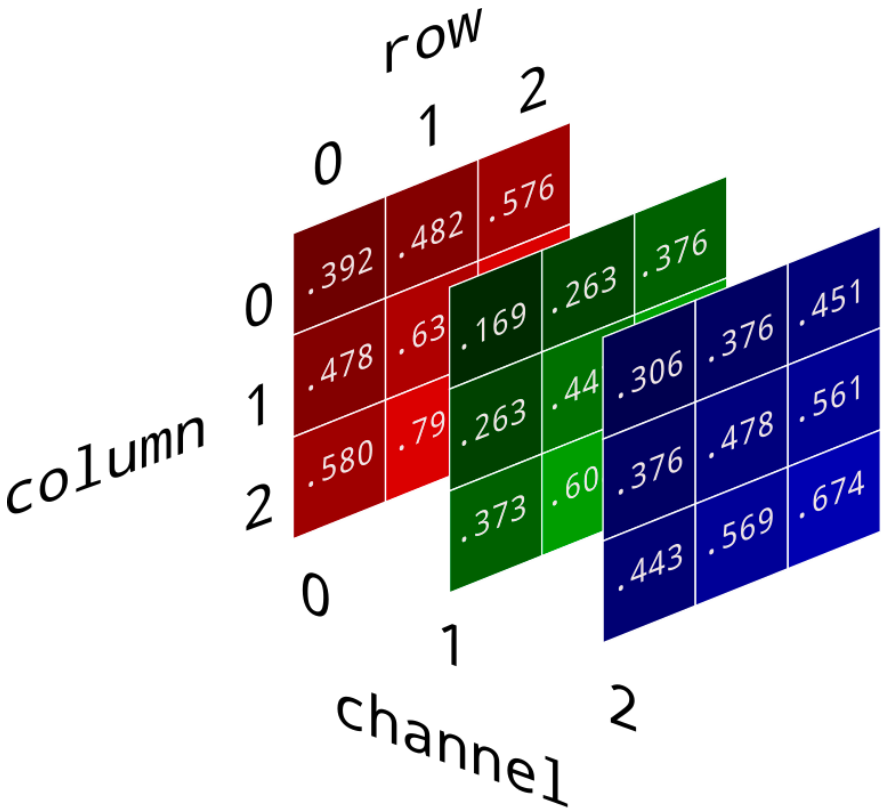
What Computer Sees

```
  0   2  15   0   0  11  10   0   0   0   0   9   9   0   0   0
  0   0   0   4  60 157 236 255 255 177  95  61  32   0   0  29
  0  10  16 119 238 255 244 245 243 250 249 255 222 103  10   0
  0  14 170 255 255 244 254 255 253 245 255 249 253 251 124   1
  2  98 255 228 255 251 254 211 141 116 122 215 251 238 255  49
 13 217 243 255 155  33 226  52   2   0  10  13 232 255 255  36
 16 229 252 254  49  12   0   0   7   7   0  70 237 252 235  62
  6 141 245 255 212  25  11   9   3   0 115 236 243 255 137   0
  0  87 252 250 248 215  60   0   1 121 252 255 248 144   6   0
  0  13 113 255 255 245 255 182 181 248 252 242 208  36   0  19
  1   0   5 117 251 255 241 255 247 255 241 162  17   0   7   0
  0   0   0   4  58 251 255 246 254 253 255 120  11   0   1   0
  0   0   4  97 255 255 255 248 252 255 244 255 182  10   0   4
  0  22 206 252 246 251 241 100  24 113 255 245 255 194   9   0
  0 111 255 242 255 158  24   0   0   6  39 255 232 230  56   0
  0 218 251 250 137   7  11   0   0   0   2  62 255 250 125   3
  0 173 255 255 101   9  20   0  13   3  13 182 251 245  61   0
  0 107 251 241 255 230  98  55  19 118 217 248 253 255  52   4
  0  18 146 250 255 247 255 255 255 249 255 240 255 129   0   5
  0   0  23 113 215 255 250 248 255 255 248 248 118  14  12   0
  0   0   6   1   0  52 153 233 255 252 147  37   0   0   4   1
  0   0   5   5   0   0   0   0   0  14   1   0   6   6   0   0
```

Color Images (RGB): Most images we see are in color. Computers typically use the RGB (Red, Green, Blue) color model to represent colors.

- Each pixel now has three values: one each for red, green, and blue.
- Each of these values ranges from 0 to 255 (or divided by 255 for values between 0-1).

- The logic is still the same the values represent the intensity of the colors
- The computer stores this as three 2D arrays (**one for each color channel**) or as a 3D array.



https://contrib.pbslearningmedia.org/WGBH/buac20/buac20-int-rgbcoloradd/index.html
https://www.csfieldguide.org.nz/en/interactives/rgb-mixer/ (these are two color playgrounds for rgb color space)

Image Resolution: The number of pixels in an image determines its resolution.

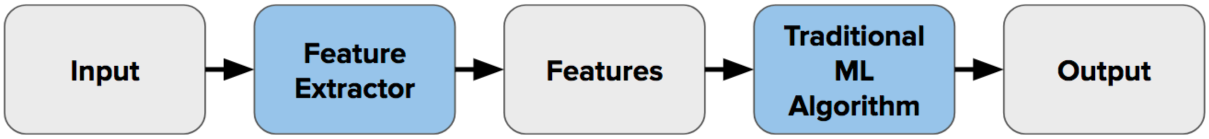
- An image that is 1000 pixels wide and 800 pixels tall has a resolution of 1000x800.
- If there is one channel (grayscale image) or three channels (RGB image) the number of pixels should match in all the channels
- Higher resolution means more detail but also larger file sizes.

MegaPixel

One million pixels. Megapixels are the measurement of the resolution of still and video cameras, monitors and scanners. For example, a 16-megapixel (16MP) still camera captures a picture composed of some 15.9 million pixels, each pixel containing a red, green and blue color dot. The image resolution is 5312x2988 (5,312 pixels across; 2,988 down).

Computer Vision

Computer vision is a field of artificial intelligence that trains computers to interpret and understand the visual world. Using images and videos, machines can identify and classify, objects, understand what going on, and react accordingly



Traditional Machine Learning Flow

Imagine you have two photos of the same scene taken from slightly different angles. Your goal is to identify the same objects or points in both images. Think of **features** as **distinctive landmarks in each photo** that can be used to identify the same points in both images.

1. Understanding Features

Features in an image are like landmarks that help us recognize parts of the image. Think of them as **points of interest** that stand out, such as **corners, edges, or unique textures**.

In our analogy, imagine you have a landscape photo and you identify features like a tree, a building, and a river bend.

2. How do we find features?

To detect features, we look for specific patterns or shapes in the image that are easy to recognize. Here's how this process works:

a. Detecting Key Points

- **Key Points**: These are **specific points in the image that stand out.**

  For example, the corners of a building, the center of a flower, or the tip of a tree.

- Imagine placing stickers on these notable spots in the image.

b. Describing Key Points

- **Descriptors**: Once we have the key points, we need to describe them in a way that helps us identify the same points in another image. Think of this as writing a small note for each sticker, describing its surroundings (e.g., "This sticker is at the corner of a building with windows nearby").

3. Matching Features

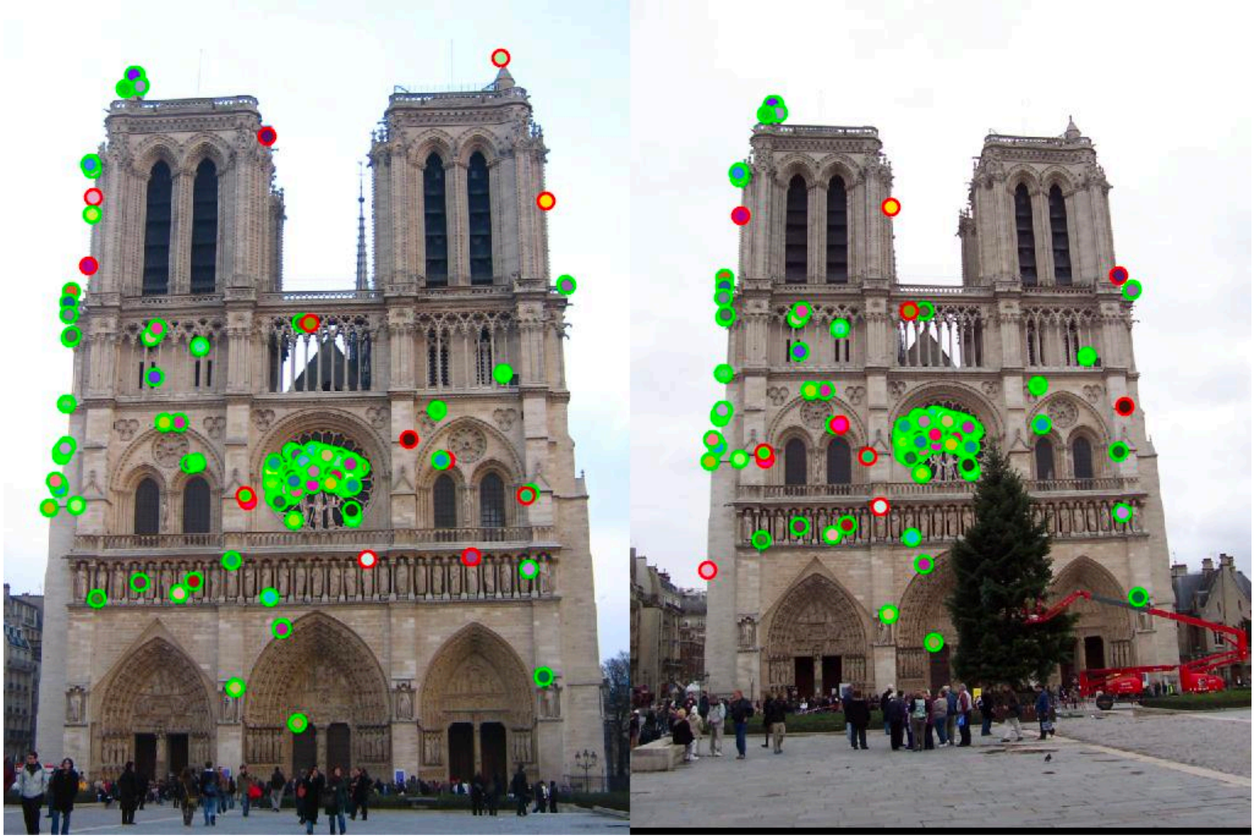
Now, you have two sets of images, each with **stickers (key points)** and **notes (descriptors)**. The goal is to match the stickers from the first image with those in the second image.

a. Comparing Descriptors

- To find matches, compare the notes (descriptors) from the first image with the notes from the second image. Look for notes that describe similar surroundings.
- For example, a note saying "corner of a building with windows nearby" in the first image should match a similar note in the second image.

b. Finding Matches

- **Matching**: If two notes (descriptors) are very similar, we consider the key points they describe to be a match. This means the stickers at those points correspond to the same landmark in both images.
- Think of this as finding that the tree in the first photo is the same tree in the second photo.



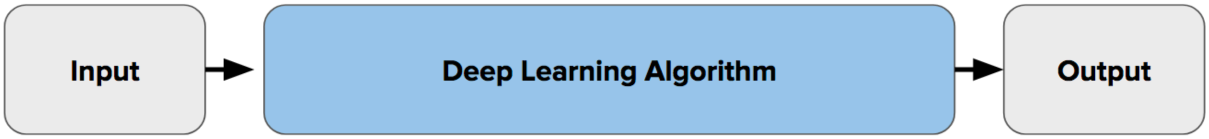
Feature extraction in computer vision is a versatile technique used across a wide range of tasks beyond just image search or similarity. Here's a brief overview of how feature extraction is applied in various computer vision tasks

- OCR algorithms use these features to recognize and interpret individual characters and words.
- Extracted features are used by object detection algorithms to identify the presence and position of objects.
- These features are tracked over time to detect motion and follow moving objects.

Intro to Opencv.ipynb

Neural Networks for Computer Vision

In recent years, neural networks have dramatically advanced the capabilities of computer vision systems, allowing machines to recognize objects, understand scenes, and even generate new images.



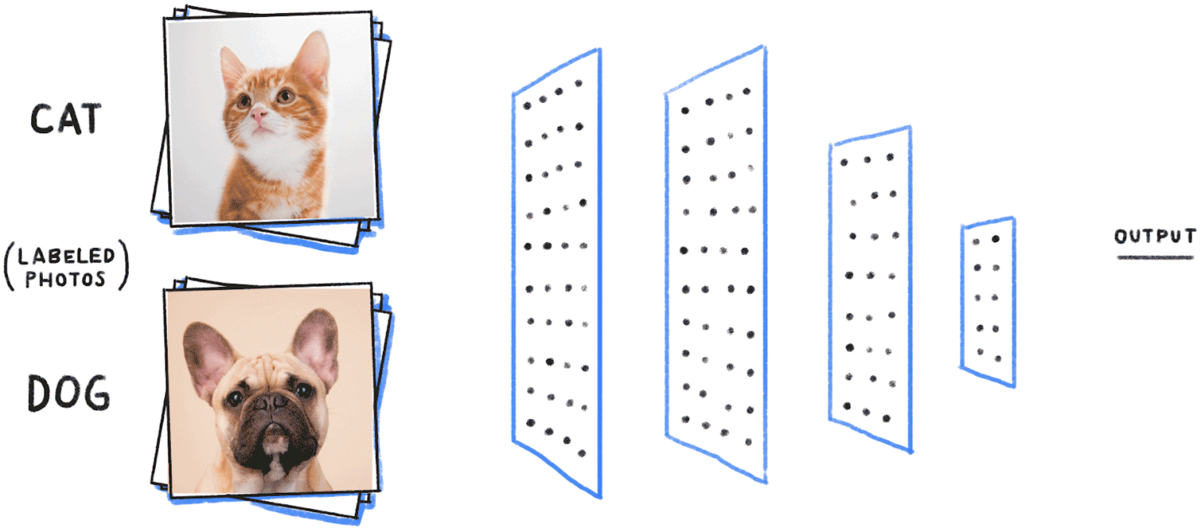
Deep Learning Flow

What All Changed

1. **Improved Accuracy**: Neural networks, especially Convolutional Neural Networks (CNNs), have drastically improved the accuracy of computer vision tasks like object detection, image classification, and image segmentation.
2. **Automated Feature Extraction**: Instead of manually selecting features, neural networks automatically learn the best features directly from the data.
3. **Handling Complexity**: Neural networks can handle more complex and varied data, making them suitable for a wide range of applications beyond the capabilities of traditional methods.
4. **Scalability**: Neural networks scale well with large datasets and can improve performance as more data becomes available.
5. **End-to-End Learning**: Neural networks allow for end-to-end learning where the input is raw data (like images) and the output is the desired result (like labels), streamlining the process.

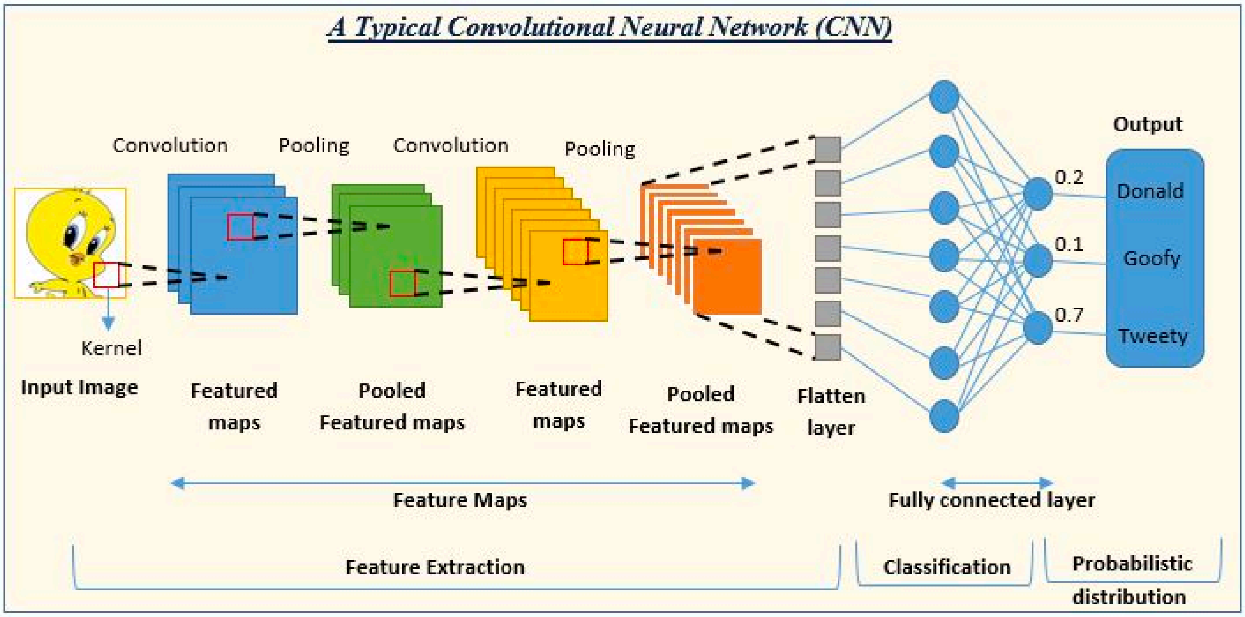
CNN

A Convolutional Neural Network (CNN) is a specialized type of neural network designed primarily for analyzing visual data, like images. While a basic neural network treats input data as a flat array of numbers, a CNN preserves the spatial structure of images by working directly with 2D or 3D data.



Lets have a brief look at how a CNN looks like and then we'll understand all the parts of this and how it works.

We'll not go into all the details but enough o give you an idea of the basics of computer vision and CNNs

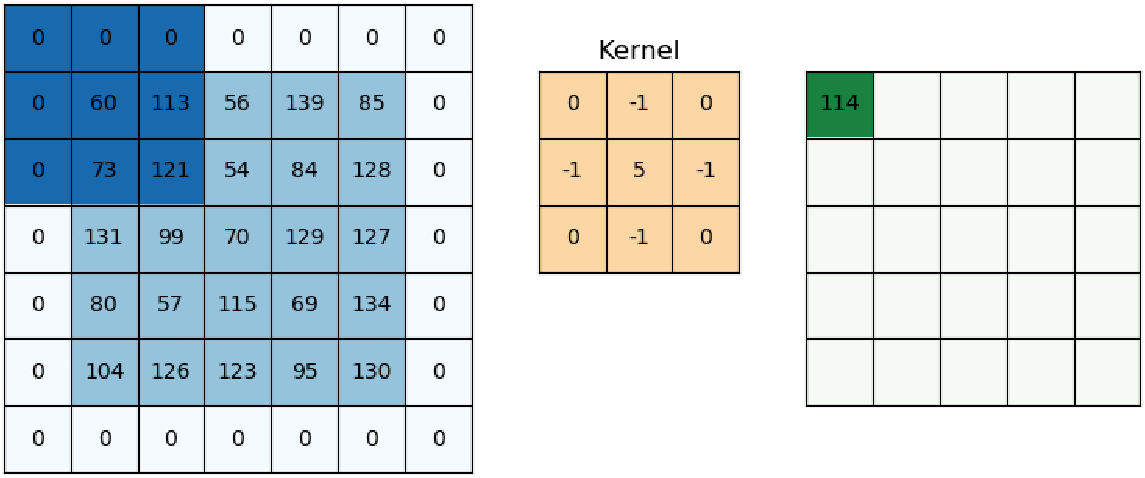


Convolutional Layers as Feature Extractors

Convolutional layers are the core components of Convolutional Neural Networks (CNNs), and their primary function is to act as feature extractors. Here's a simple explanation of how they work:

The Process of Convolution

- **Feature Detection (Convolutions)**: Imagine you have different colored glasses that help you see certain things better, like red glasses for finding all the red parts in a picture. Similarly, CNN uses different filters to find different features in the image. For example, **one filter might detect edges, another might find corners, and another might recognize textures.**
    - Each filter is like a small window that scans through the entire picture, looking for specific features.
        - Each filter is a small matrix of numbers (e.g., 3x3, 5x5)
        - The weights on each cell of the filter / kernel represent the multiplication factor for the corresponding data point on the area it is overlapping on the image, and are used to calculate the value of corresponding data point on the matrix of convolution features.
    - A convolutional layer uses small, learnable filters (or kernels) that move across the input image.
    - Stride: This refers to the number of pixels by which the filter moves at each step. A stride of 1 means the filter moves one pixel at a time.
    - Padding: Sometimes, zeroes are added around the border of the image so that the filter can fit perfectly over the edges. This helps preserve the spatial dimensions of the image after convolution.



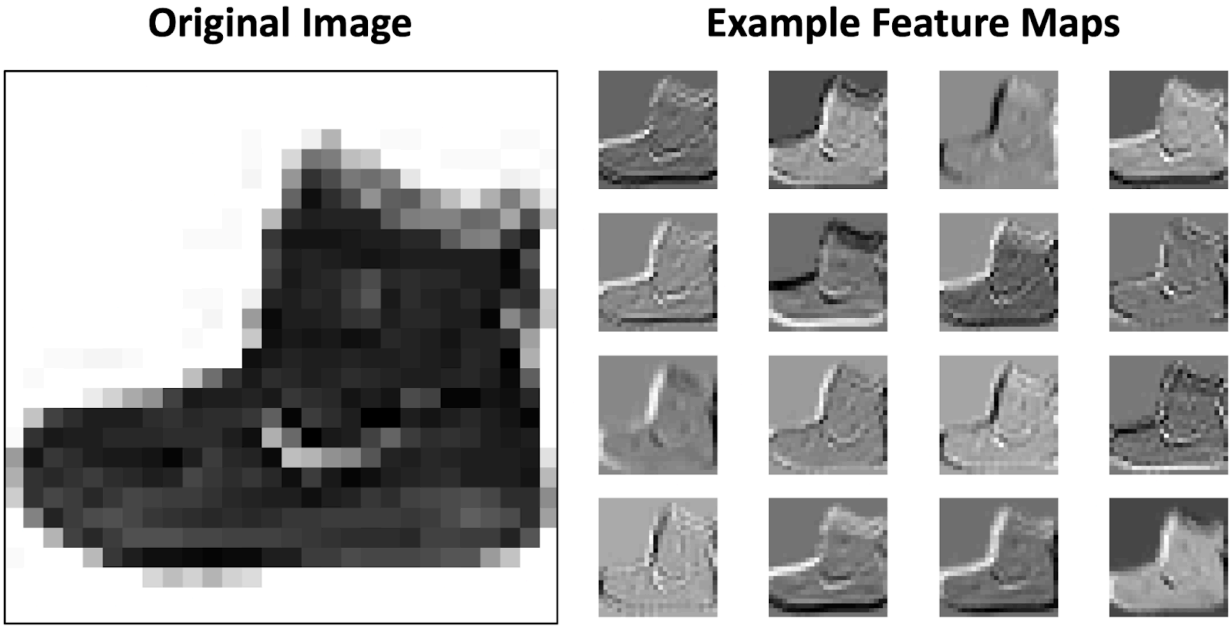
0	0	0	0	0	0	0
0	60	113	56	139	85	0
0	73	121	54	84	128	0
0	131	99	70	129	127	0
0	80	57	115	69	134	0
0	104	126	123	95	130	0
0	0	0	0	0	0	0

Kernel

0	-1	0
-1	5	-1
0	-1	0

114				

- **Creating a Feature Map**: As the filters scan across the image, they **create a new, smaller image called a feature map**. This feature map **highlights where certain features (like edges or textures) are found in the original image**. Think of this as creating a map that shows where all the interesting parts of the picture are located.
  - At each position, the filter multiplies its values by the corresponding pixel values of the image and sums up the results.



Original Image

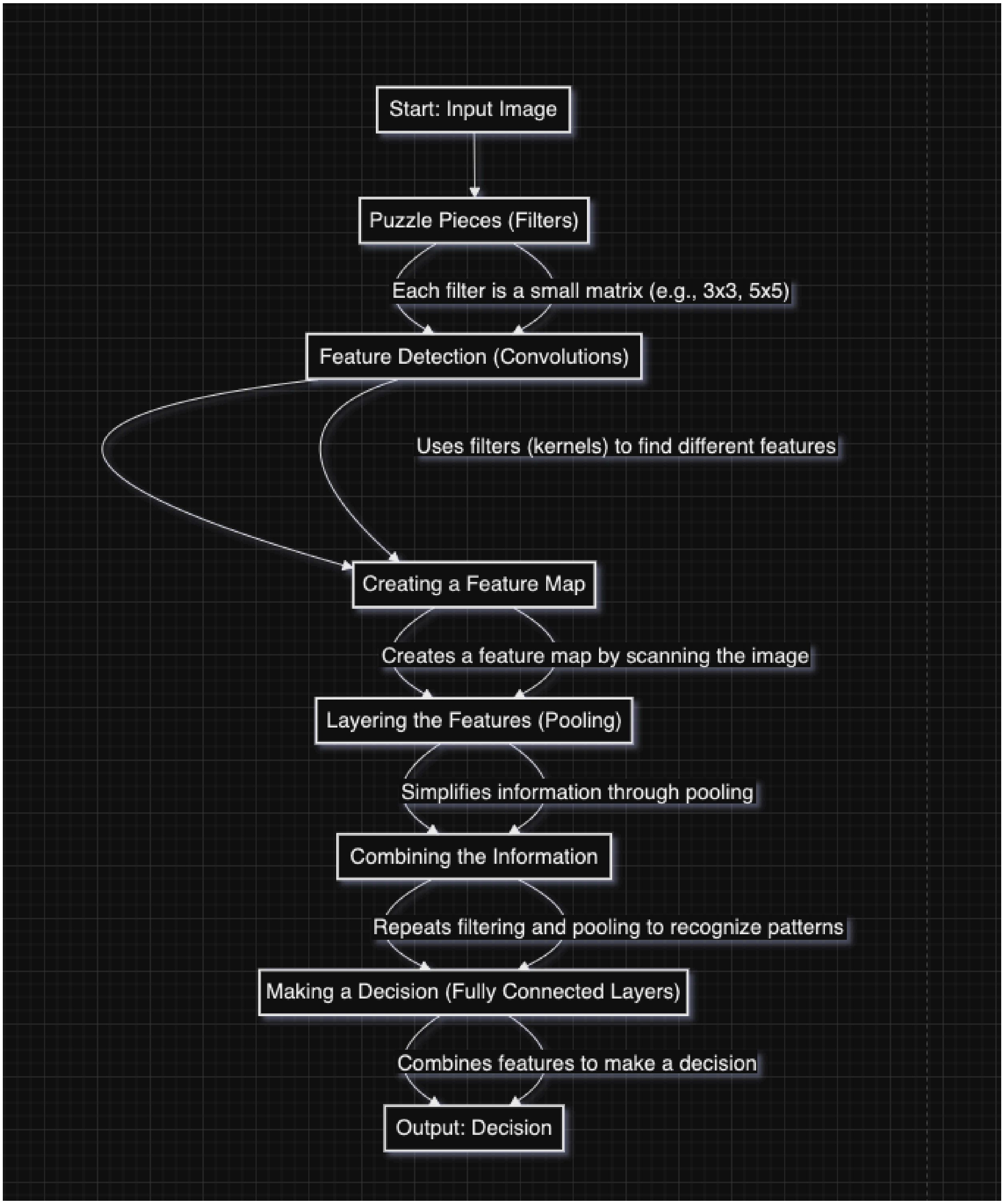
Example Feature Maps

- **Layering the Features (Pooling)**: After creating feature maps, the CNN simplifies the information. Imagine looking at the puzzle through a magnifying glass that only shows **the most important parts**. This is done through pooling, where the CNN **reduces the size of the feature maps while keeping the important details**. It's like summarizing the features to make it easier for the computer to understand.

https://www.youtube.com/watch?v=f1fXCRtSUWU

- **Combining the Information**: The CNN repeats this process of filtering and pooling several times, each time looking at more complex patterns. Initially, it might find simple things like edges, then more complex patterns like shapes, and finally, it recognizes high-level features like eyes, noses, or fur patterns.
- **Making a Decision (Fully Connected Layers)**: Finally, all the gathered and simplified information is put together to make a decision. Imagine you have a checklist for identifying a cat or a dog. The CNN uses the combined features to compare against this checklist and decide what the picture shows

.


Start: Input Image
Puzzle Pieces (Filters)
Each filter is a small matrix (e.g., 3x3, 5x5)
Feature Detection (Convolutions)
Uses filters (kernels) to find different features
Creating a Feature Map
Creates a feature map by scanning the image
Layering the Features (Pooling)
Simplifies information through pooling
Combining the Information
Repeats filtering and pooling to recognize patterns
Making a Decision (Fully Connected Layers)
Combines features to make a decision
Output: Decision

Hierarchical Feature Extraction

1. **Low-Level Features**
    - **First Layer**: The initial convolutional layer typically learns to detect basic features like edges, corners, and simple textures. These are low-level features that are common in most images.
2. **Mid-Level Features**
    - **Subsequent Layers**: As the data passes through more convolutional layers, the network starts to combine the low-level features to detect more complex patterns and shapes (e.g., circles, squares, textures).
3. **High-Level Features**
    - **Deeper Layers**: In the deeper layers, the network combines these mid-level features to recognize high-level features and objects (e.g., faces, animals, cars). This hierarchical learning allows the network to build a detailed and abstract representation of the input image.

https://poloclub.github.io/cnn-explainer/ (this is a great site with visualisations to explain cnn working) on the top of the website the image is interactable.

Intro to Opencv.ipynb