Agenda:

- Computer vision concepts
- Computer vision capabilities in Azure

Computer Vision

Computer vision is one of the core areas of artificial intelligence (AI), and focuses on creating solutions that enable AI applications to "see" the world and make sense of it.

Of course, computers don't have biological eyes that work the way ours do, but they're capable of processing images; either from a live camera feed or from digital photographs or videos. This ability to process images is the key to creating software that can emulate human visual perception.

In this module, we'll examine some of the fundamental principles and techniques that underly computer vision. We'll also introduce Microsoft Azure AI Vision, a cloud service that developers can use to create a wide range of computer vision solutions.

Images and image processing

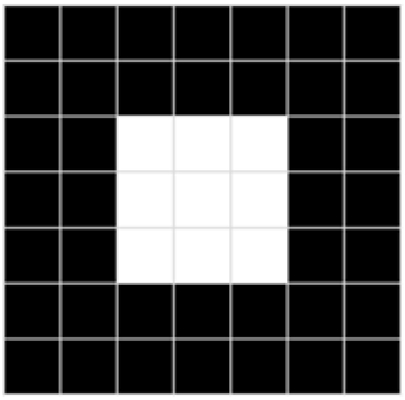
Before we can explore image processing and other computer vision capabilities, it's useful to consider what an image actually *is* in the context of data for a computer program.

Images as pixel arrays

To a computer, an image is an array of numeric pixel values. For example, consider the following array:

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	255	255	255	0	0
0	0	255	255	255	0	0
0	0	255	255	255	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

The array consists of seven rows and seven columns, representing the pixel values for a 7x7 pixel image (which is known as the image's *resolution*). Each pixel has a value between 0 (black) and 255 (white); with values between these bounds representing shades of gray. The image represented by this array looks similar to the following (magnified) image:



The array of pixel values for this image is two-dimensional (representing rows and columns, or *x* and *y* coordinates) and defines a single rectangle of pixel values. A single layer of pixel values like this represents a grayscale image. In reality, most digital images are multidimensional and consist of three layers (known as *channels*) that represent red, green, and blue (RGB) color hues. For example, we could represent a color image by defining three channels of pixel values that create the same square shape as the previous grayscale example:

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	255	255	255	0	0
0	0	255	255	255	0	0
0	0	255	255	255	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

Convolutional Neural network (CNN)

Imagine you're a detective looking at a crime scene photo (the image).

1. **The magnifying glass (filter) scans the scene:**
    - You have a magnifying glass (the filter) that highlights specific details (edges, shapes, colors).
    - You systematically move the magnifying glass across the photo, picking up on these details.
2. **Making a sketch (feature map):**
    - Each time you find an interesting detail, you sketch it on a notepad (the feature map).
    - This notepad holds different sketches depending on what the magnifying glass highlights (edge map, color map, etc.).
3. **Noting the most important clues (pooling):**

- You don't want to be overwhelmed by every tiny detail, so you summarize the key points from each sketch (pooling).
- You might pick the most prominent edge or the most vibrant color in a specific area.

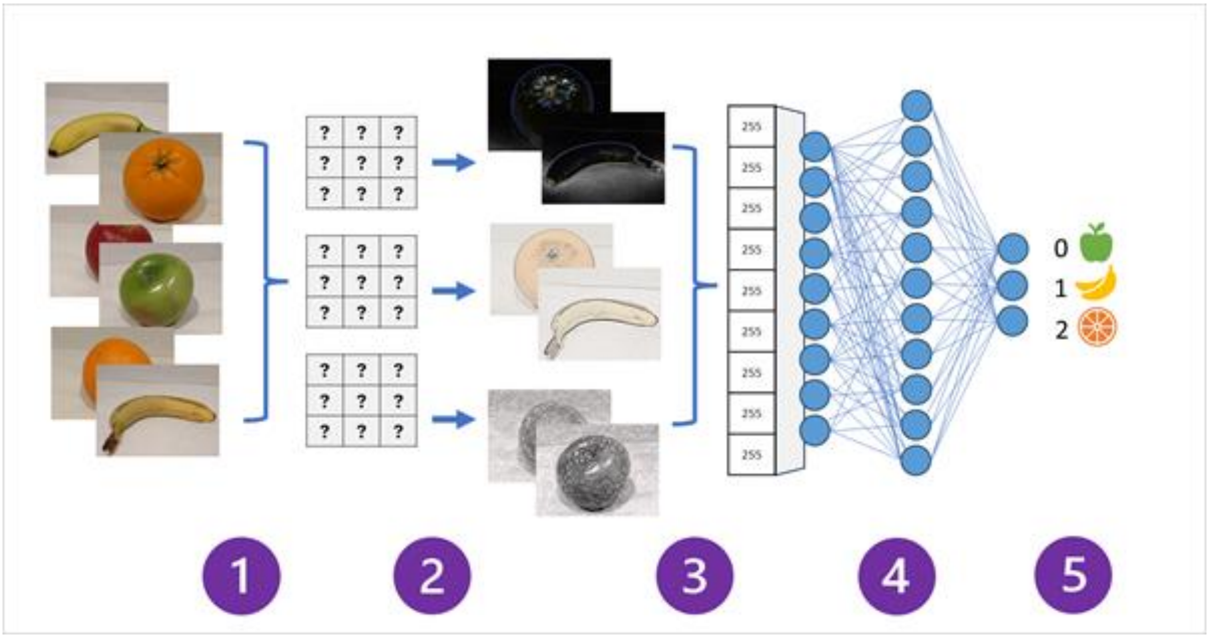
4. **Connecting the clues (fully-connected layers):**
    - After examining all areas with the magnifying glass, you piece together the clues (features) to form a conclusion (fully-connected layers).
    - Based on the edges, shapes, and colors, you might determine there's a dog in the photo.

CNNs do something similar, but with millions of photos and much faster!

- They have many magnifying glasses (filters) that look for various features.
- They create many sketchpads (feature maps) to hold these different features.
- They summarize the important details from each sketchpad (pooling).
- Finally, they connect the features to make a final decision (classification).

Benefits of CNNs (like a good detective):

- **Learn automatically (no need to pre-teach):** The detective doesn't need to be told what edges or colors to look for - they learn from experience (training data). CNNs do the same.
- **Focus on the big picture:** By summarizing the details, the detective can focus on what truly matters. Pooling in CNNs achieves a similar effect.



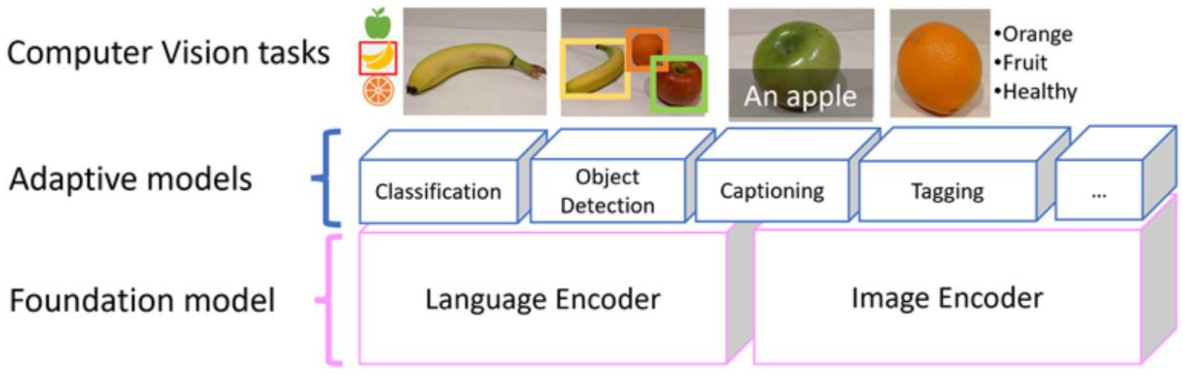
One of the most common machine learning model architectures for computer vision is a *convolutional neural network* (CNN). CNNs use filters to extract numeric feature maps from images, and then feed the feature values into a deep learning model to generate a label prediction. For example, in an *image classification* scenario, the label represents the main subject of the image (in other words, what is this an image of?). You might train a CNN model with images of different kinds of fruit (such as apple, banana, and orange) so that the label that is predicted is the type of fruit in a given image.

During the *training* process for a CNN, filter kernels are initially defined using randomly generated weight values. Then, as the training process progresses, the models predictions are evaluated against known label values, and the filter weights are adjusted to improve accuracy. Eventually, the trained fruit image classification model uses the filter weights that best extract features that help identify different kinds of fruit.

The following diagram illustrates how a CNN for an image classification model works:

1. Images with known labels (for example, 0: apple, 1: banana, or 2: orange) are fed into the network to train the model.
2. One or more layers of filters is used to extract features from each image as it is fed through the network. The filter kernels start with randomly assigned weights and generate arrays of numeric values called *feature maps*.
3. The feature maps are flattened into a single dimensional array of feature values.
4. The feature values are fed into a fully connected neural network.
5. The output layer of the neural network uses a *softmax* or similar function to produce a result that contains a probability value for each possible class, for example [0.2, 0.5, 0.3].

Multi-modal models



The Microsoft *Florence* model, which is used for Image Analysis 4.0 capabilities in Azure AI Vision, is a multi-modal model. Trained with huge volumes of captioned images from the Internet, it includes both a language encoder and an image encoder. Florence is an example of a *foundation* model. In other words, a pre-trained general model on which you can build multiple *adaptive* models for specialist tasks. For example, you can use Florence as a foundation model for adaptive models that perform:

- *Image classification*: Identifying to which category an image belongs.
- *Object detection*: Locating individual objects within an image.

- *Captioning*: Generating appropriate descriptions of images.
- *Tagging*: Compiling a list of relevant text tags for an image.