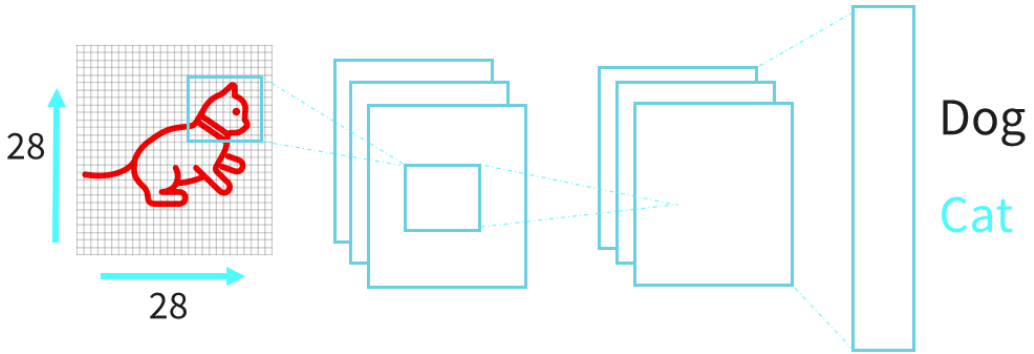
Convolution & Pooling

Holberton

Why convolution & pooling in machine learning

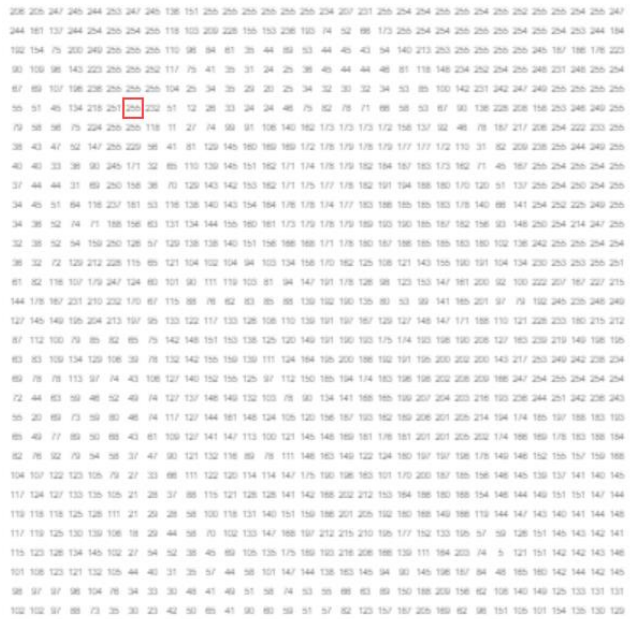
Why Convolution & Pooling for Machine Learning

- Building blocks of Convolutional Neural Networks
  - Widely used in image recognition, object detection
- Convolution helps in feature extraction
- Pooling helps to downsample data



Background knowledge: image kernels

- Image kernel = small matrix of data
  - Commonly used in edge detection, blurring, or sharpening filters



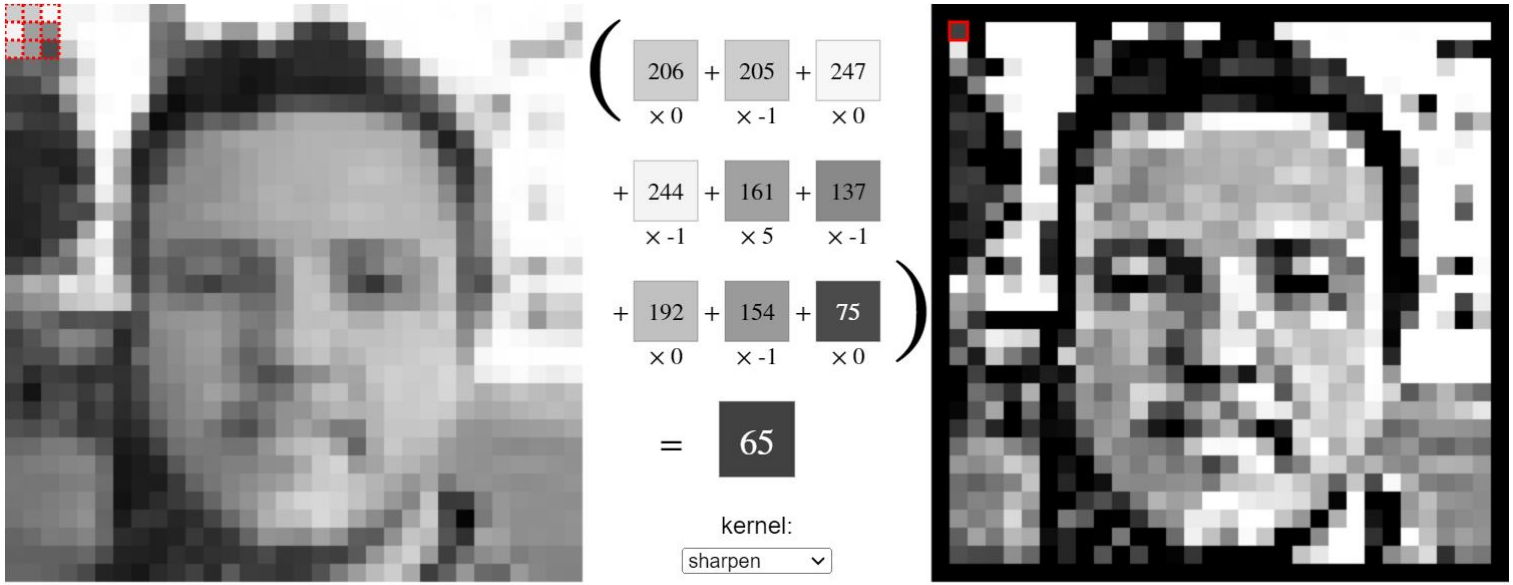
What is convolution?

What is convolution

- Convolution is a mathematical operation where a filter slides over input data
  - It computes the **element-wise** multiplication between the filter and input at each position
- Convolution is used in feature extraction
  - Input data is usually a matrix representing an image
- Filter is a smaller matrix that slides over the input to extract features
  - The element-wise multiplication results in a new matrix called **feature map**

What is convolution

- Convolution is a mathematical operation where a filter slides over input data

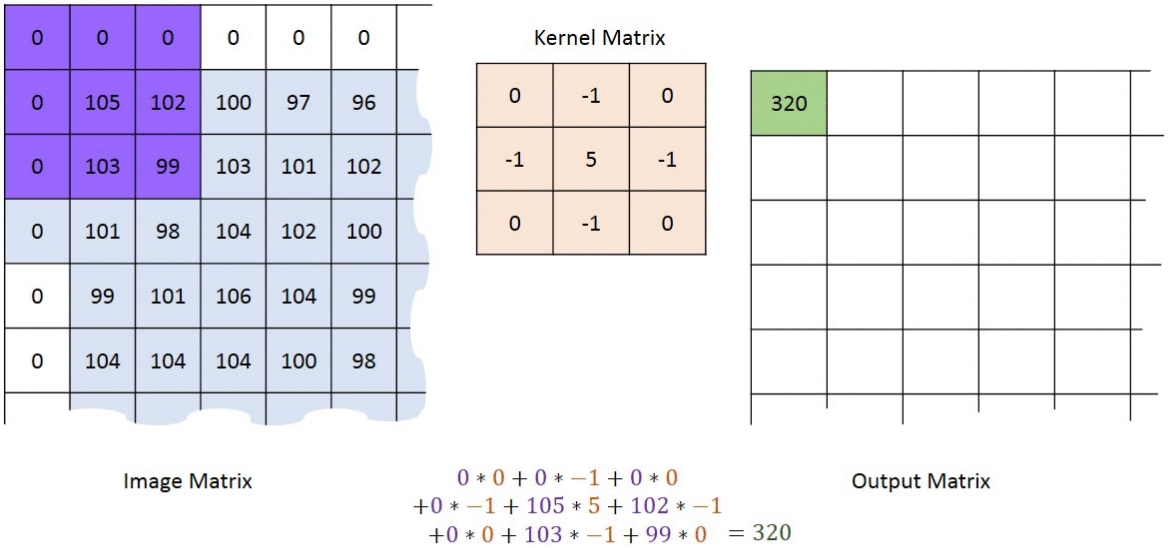


input image

output image

What is convolution

- Convolution is a mathematical operation where a filter slides over input data



Convolution with horizontal and vertical strides = 1

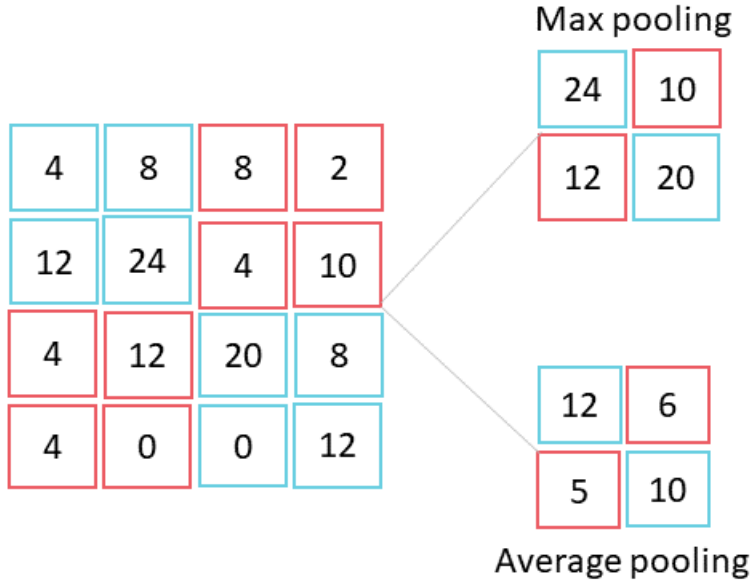
What is pooling?

What is pooling

- Pooling refers to a down-sampling process
  - Reduces the spatial dimensions of the feature maps produced by convolution
- Pooling is typically applied after a convolution layer
- Pooling helps to prevent “overfitting„
  - Reduces the dimensionality of the feature map & parameters of the model
- Pooling makes the model more robust to small variations
  - ...but can result in loss of information

Types of pooling

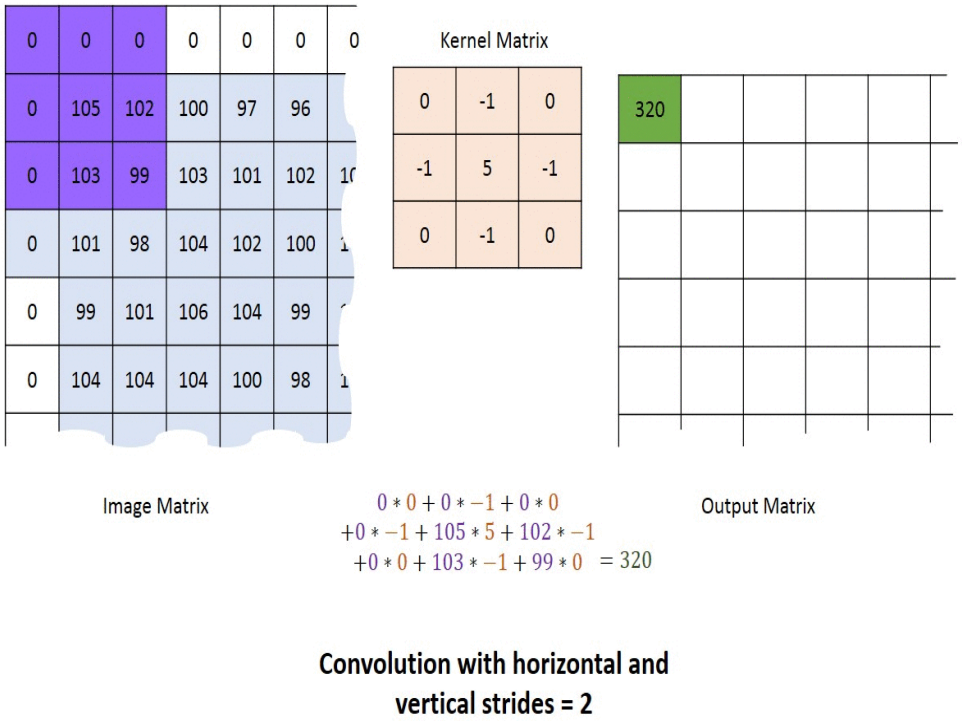
- Pooling divides the input image / feature map into small non-overlapping regions
  - ...then it replaces each region with a single value
- **Max** pooling:
  - takes maximum value in region
- **Average** pooling:
  - takes average value in region



Fundamental components of convolution & pooling

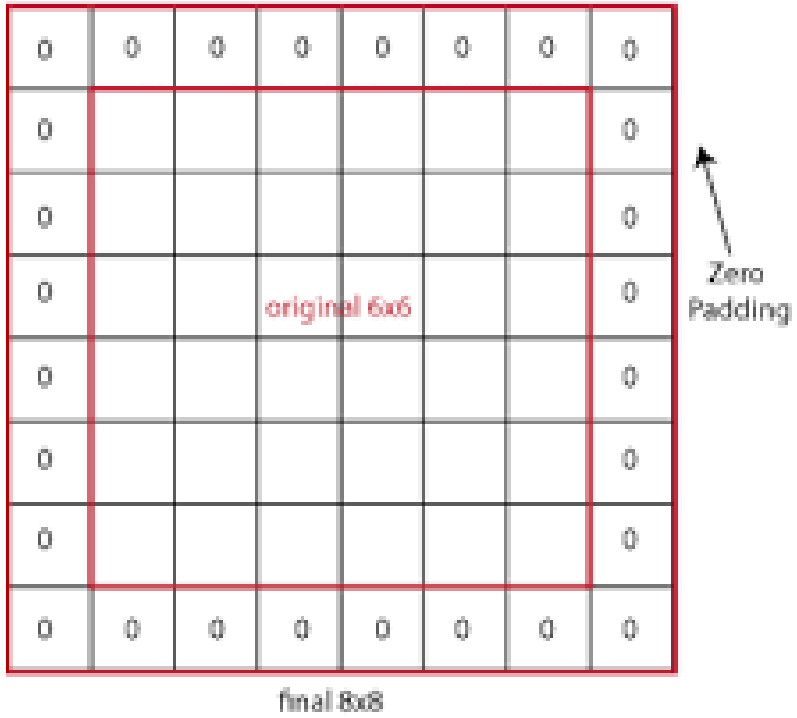
Fundamental elements: stride

- **Stride** = number of pixels by which the kernel slides over the input
- A stride of 1 means the kernel moves one pixel at at time
- Increasing the stride reduces the spatial dimensions
- Smaller stride provides fine-grained output feature map



Fundamental elements: padding

- **Padding** = ensure feature map has same dimensions as input
- Padding adds extra rows and cols of zeroes around the edge of input image or feature map
- When no padding is used, the kernel may not slide all the way to the edge

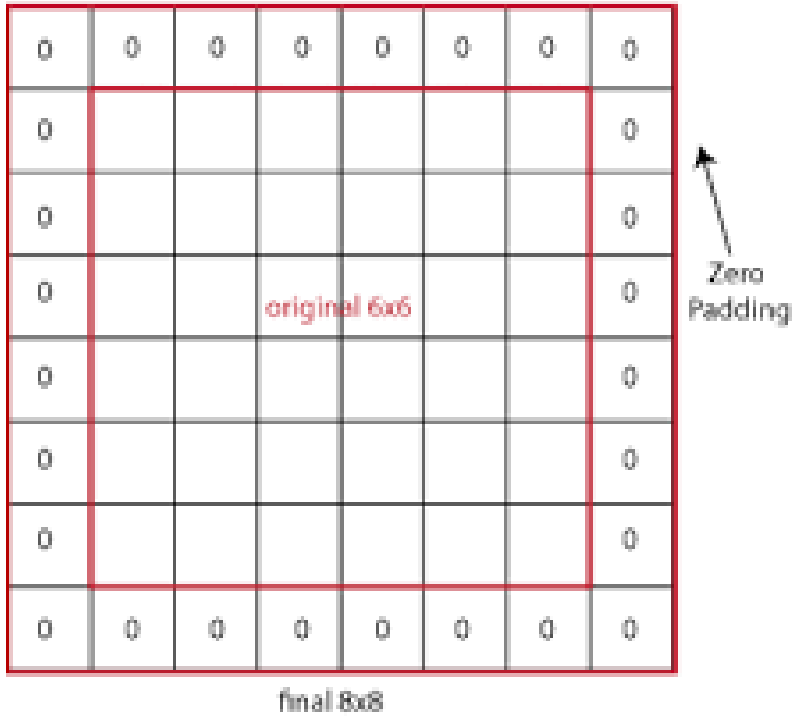


Fundamental elements: types of padding

- **Zero padding:** zeroes are added around the edge of the input image

- **Same padding:** padding is added to input to match spatial dimensions of output

- Many other types of padding exist

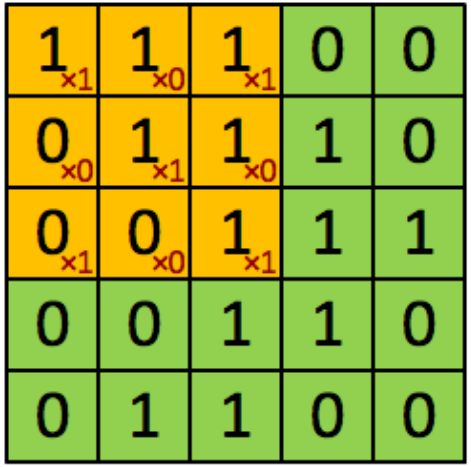


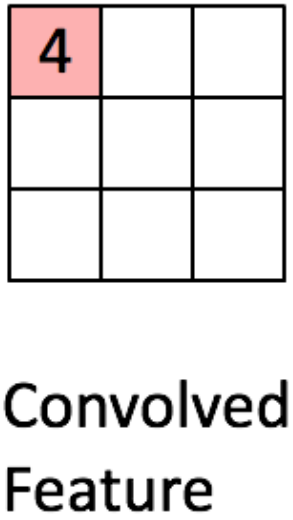
Fundamental elements: feature map

- **Feature map** = 3D array of values representing the output of a convolutional layer

- Each value corresponds to a particular region in input image

- Number of feature maps corresponds to number of kernels





Convolution over an image

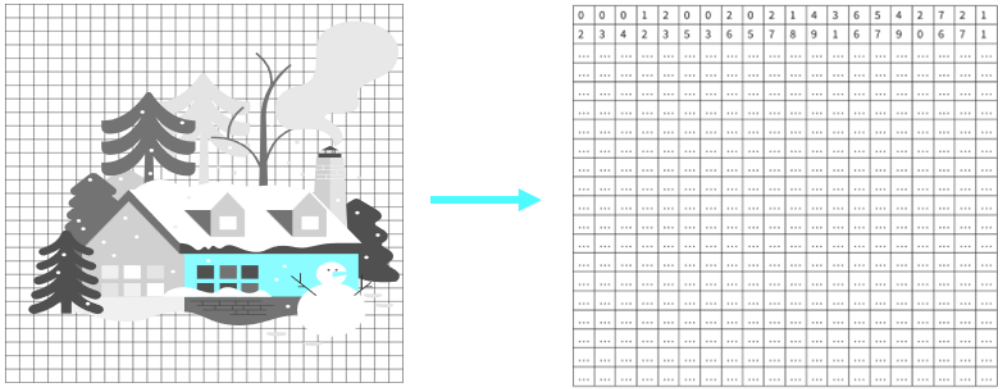
How to detect objects in images?

- Images are represented as layers of pixel values
- Rely on convolution operations to detect features
- Rely on pooling operations to reduce complexity

How to detect objects in images?

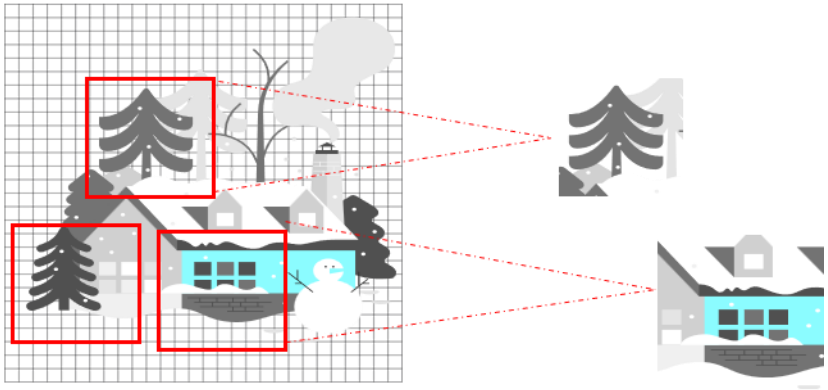
- Images are represented as layers of pixel values

- Each pixel denotes the value of a color channel (0..255)



How to detect objects in images?

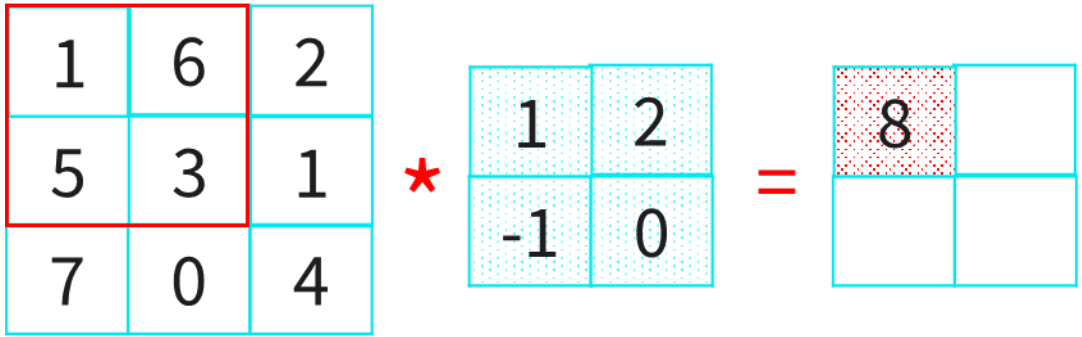
- The convolution layer



- Mathematical operation on input
  - Relies on input, filter & feature map

How to detect objects in images?

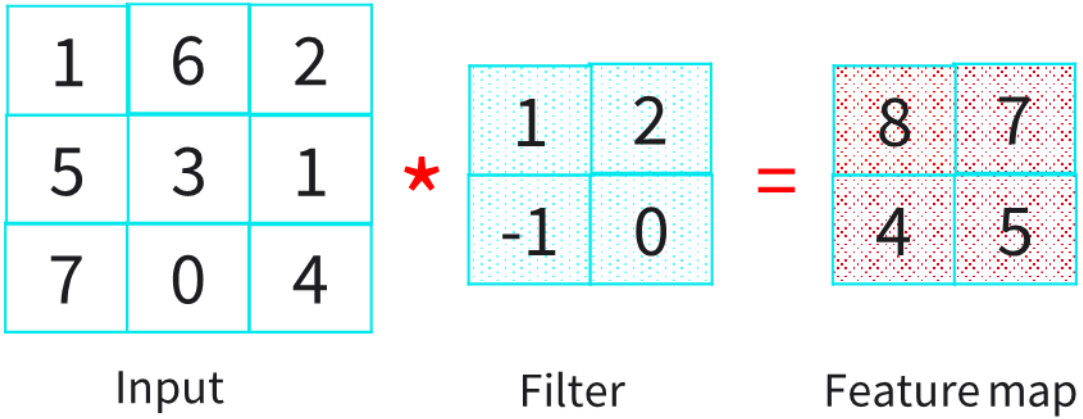
- Convolution & filter
- Example: $1 \cdot 1 + 2 \cdot 6 + (-1) \cdot 5 + 0 \cdot 3 = 8$



- Filter moves across an image and checks if a feature is present

How to detect objects in images?

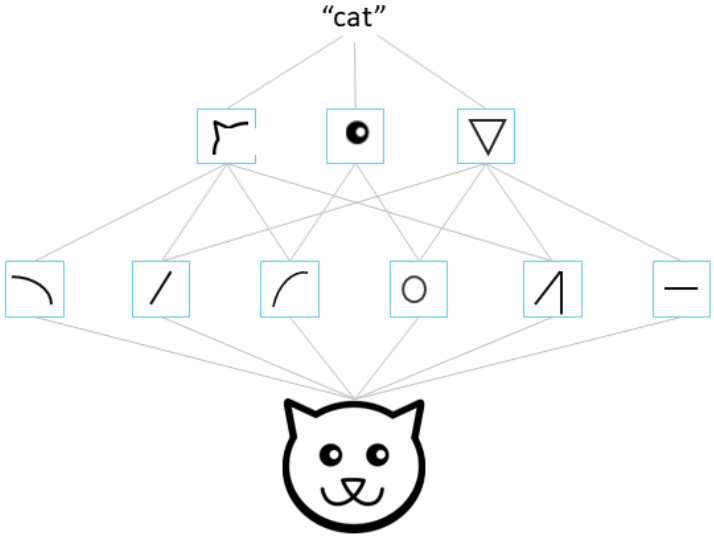
- Convolution & filter



- Filter moves across an image and checks if a feature is present

How to detect objects in images?

- Visual spatial hierarchy example



- Patterns not affected by rotation & translation
- Builds up layer of abstraction

Any questions?