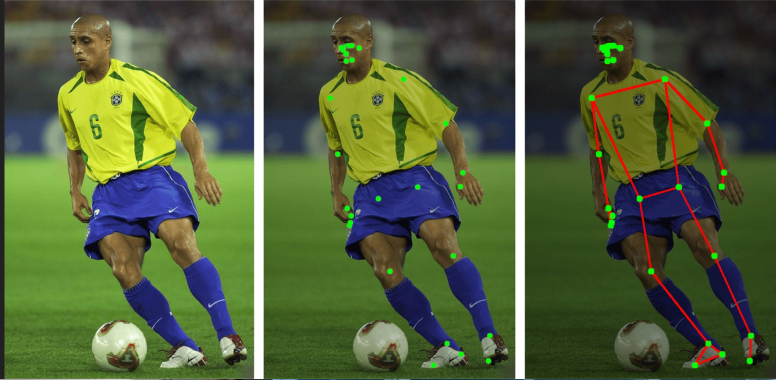
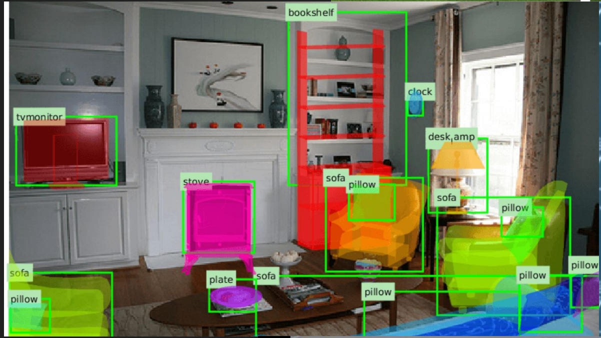
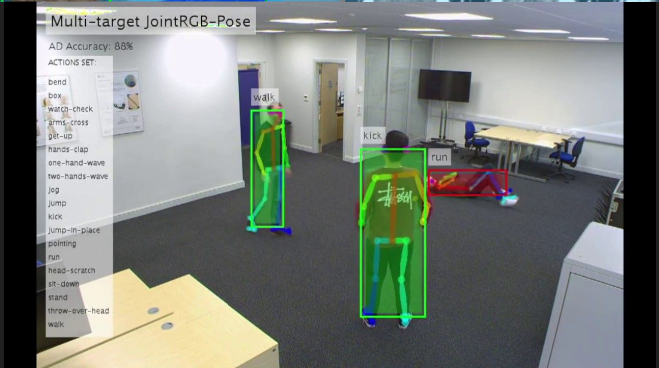
Intro to Computer Vision

What is Computer Vision?

- Broadly, a field of computer science focused on understanding images and videos
- Subfields include:
  - Scene reconstruction
  - Object detection
  - Object recognition
  - Activity recognition
  - Pose estimation
  - Motion estimation
  - …and much more!
- Where is it used?
  - Almost everywhere!
  - Healthcare: Medical imaging analysis, disease detection
  - Transportation: Self-driving vehicles, license plate recognition
  - Entertainment: Face filters, AR/VR

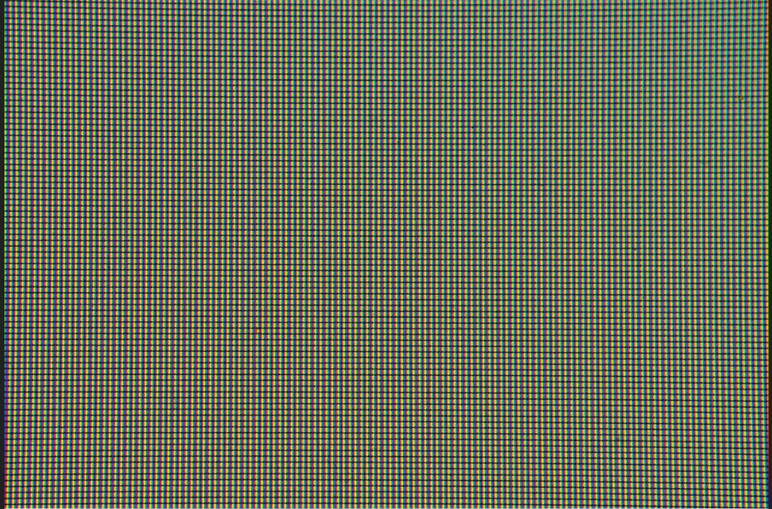


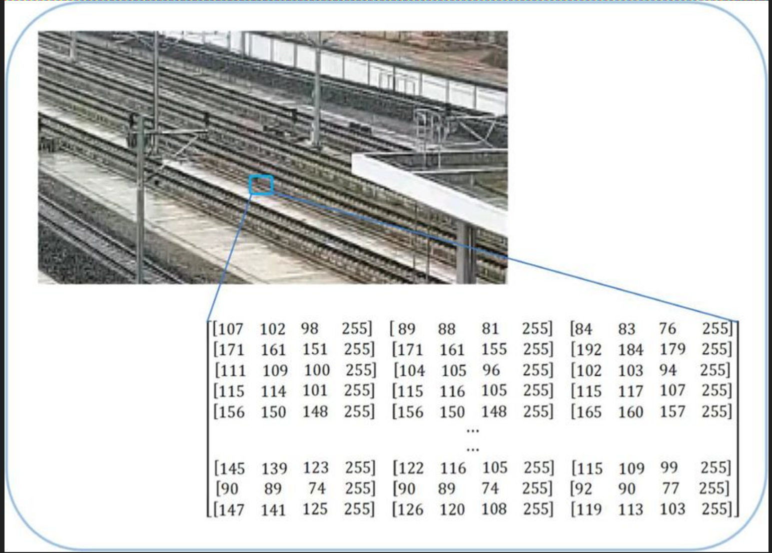




Imaging Basics

- A pixel: Smallest unit of a digital image
- Resolution: Number of pixels in an image (width x height)
- Color depth: Bits needed to represent each pixel(bpp)
  - 1 bit: (0-1)
  - 8-bit (0-255)
  - 24-bit (RGB)
  - 32-bit (RGBA, supports transparency)
- Color spaces:
  - RGB
  - BGR
  - Grayscale
- HSV(Bettering Human Perception)
  - Color type, Saturation, Value (Brightness)
- Most computers store images as **matrices**
  - 2D array for grayscale images
  - 3D array for color images
  - Each element represents one pixel

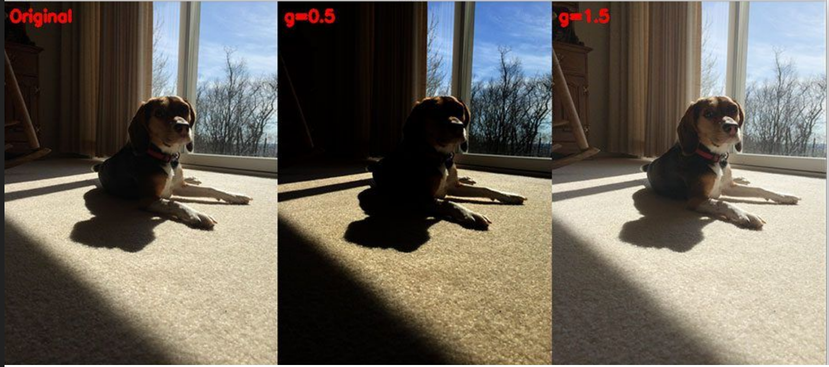


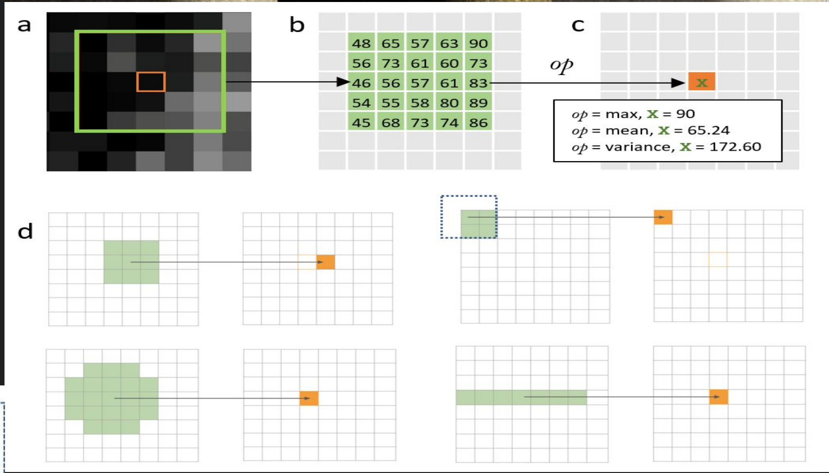


```
[[107  102  98   255]  [89   88   81   255]  [84   83   76   255]
 [171  161  151  255]  [171  161  155  255]  [192  184  179  255]
 [111  109  100  255]  [104  105  96   255]  [102  103  94   255]
 [115  114  101  255]  [115  116  105  255]  [115  117  107  255]
 [156  150  148  255]  [156  150  148  255]  [165  160  157  255]
                                ...
                                ...
 [145  139  123  255]  [122  116  105  255]  [115  109  99   255]
 [90   89   74   255]  [90   89   74   255]  [92   90   77   255]
 [147  141  125  255]  [126  120  108  255]  [119  113  103  255]]
```

Basic Image Processing

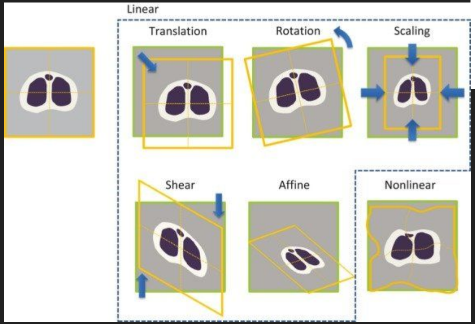
1. Point Operations
   a. Brightness adjustment (+ or - value)
   b. Contrast enhancement (multiplication)
   c. Thresholding (binary conversion)
   d. Gamma correction (power law)
2. Geometric Transformations
   a. Scaling (resize)
   b. Rotation (angle)
   c. Translation (move)
   d. Flipping (mirror)
   e. Affine transformations
3. Neighborhood Operations
   a. Smoothing (blur)
   b. Sharpening
   c. Edge detection
   d. Noise reduction
   e. Median filtering





48	65	57	63	90
56	73	61	60	73
46	56	57	61	83
54	55	58	80	89
45	68	73	74	86

op = max, **x** = 90
op = mean, **x** = 65.24
op = variance, **x** = 172.60



What we will be using today

- **Google Colab**: For sharing code
- **Python:** Programming language commonly used for machine learning
- **OpenCV:** Library that provides functions for real-time and offline computer vision
- **PyTorch:** Deep learning library used for constructing neural networks
- **Torchvision:** PyTorch extension containing functions for image transformations and augmentations
- **Matplotlib:** For visualizing our data

Notebook With Exercises

https://tinyurl.com/uclacvdatateach

Exercise 1: Loading and basic image processing

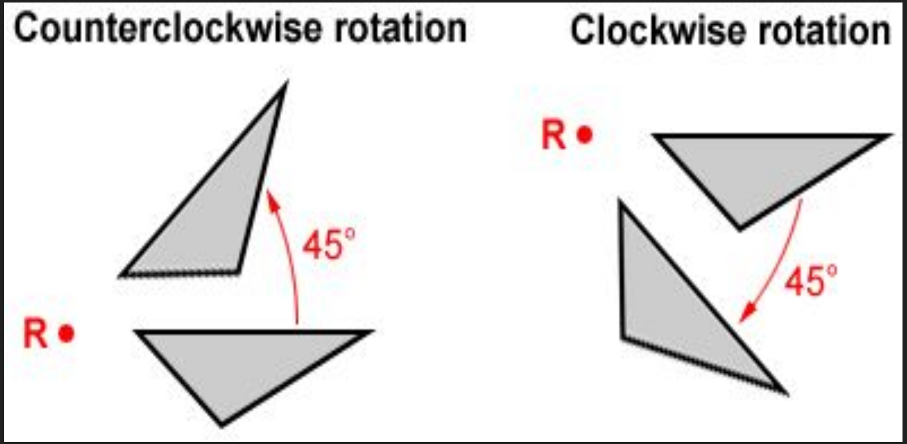
- Since most image operations happen on a matrix representation of the image, our first step is to “read” the image as matrix
- We can do this with cv2.imread(“/path/to/image”)
- For efficiency, OpenCV converts an image to a **Numpy array**, which is more efficient than a Python list
    - You can get the data type of a Numpy array with .dtype
    - You can get the max and min with .max() and .min() respectively

- Since color images are represented by their red, green, and blue components, we can break apart images and “stitch” them back together

Exercise 2: Basic Image Operations

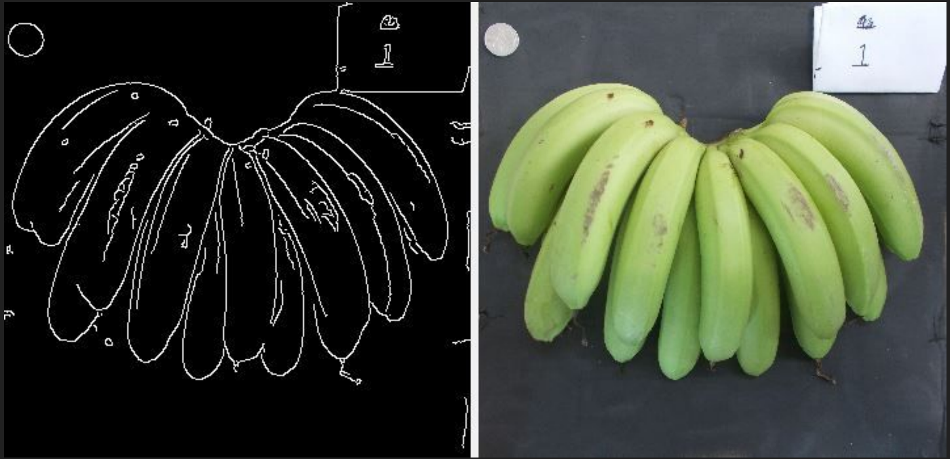
- Resizing
  - Involves scaling the dimensions of an image
  - It preserves the original proportions
- Cropping
  - Extracts a specific portion of the image
  - Removes unwanted outer areas
  - Helps focus on the main subject
- Rotation
  - Turns the image around its center point
  - Can be done in degrees (90°, 180°, 270°)
  - May require padding or cropping to handle corners
- Flipping
  - Mirrors the image horizontally or vertically
  - Useful for data augmentation
  - Helps models learn orientation invariance

- cv2.resize(image, (new_width, new_height), interpolation)
  - Takes in three parameters: image to resize, (width, height), interpolation



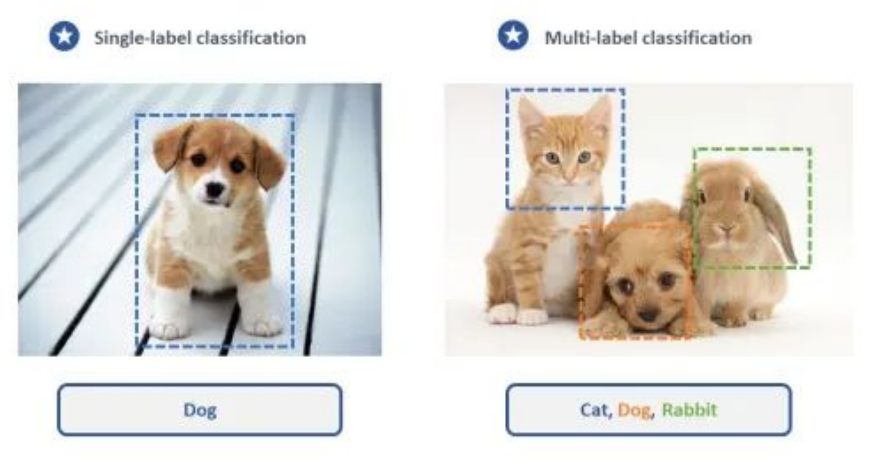
Exercise 3: Landmark Detection

- Gaussian Blur
  - Reduces image noise and detail
  - Uses a Gaussian kernel (bell-shaped) to “smooth” out an image
  - Parameters:
    - Size of the kernel
    - Kernel intensity
    - Essentially preprocessing
- Contour Drawing
  - Finds areas of intensity in the image
  - Find contours (cv2.findContours)
  - Draw contours (cv2.drawContours)
  - Applications
    - Object counting
    - Shape detection
    - Feature extraction
- Background Subtraction
  - Thresholding (cv2.threshold)
  - Creates a binary mask and applies it to the image
  - Used for: isolating objects, removing unwanted objects, preparing images for contour drawing



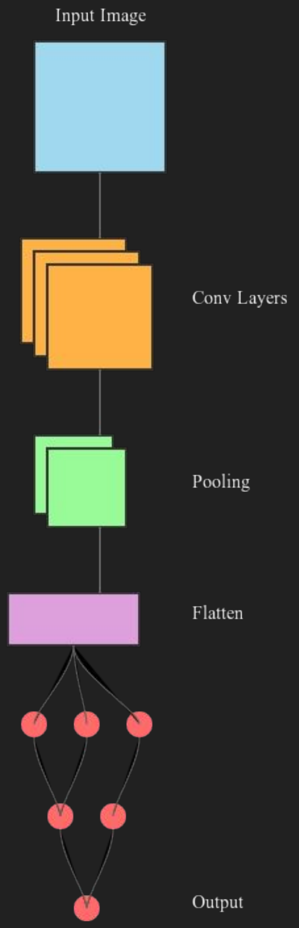
Machine Learning in Computer Vision

- Machine Learning: Process of teaching computers to learn from data without explicit programming
  - System learns patterns and features from examples instead of following rules
- Very useful for images, where data is often unstructured and is difficult to consider all scenarios
- ML can
  - Handle variations in lighting, pose, scale
  - Learn relevant features automatically
  - Generalize to new, unseen examples
  - Adapt to different conditions
- Types of Training data
  - Labeled images for supervised learning
  - Unlabeled images for unsupervised learning
- Use cases
  - Classification tasks (object recognition)
  - Detection tasks (bounding boxes)
  - Segmentation tasks (pixel-level labels)
  - Can require significant human annotation effort



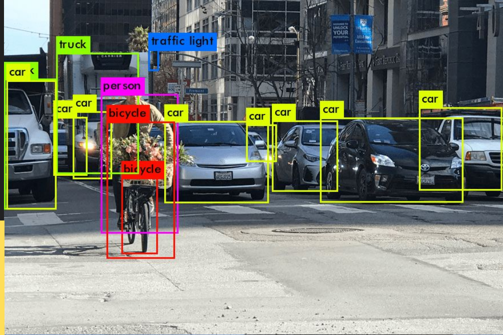
Deep Learning in Computer Vision

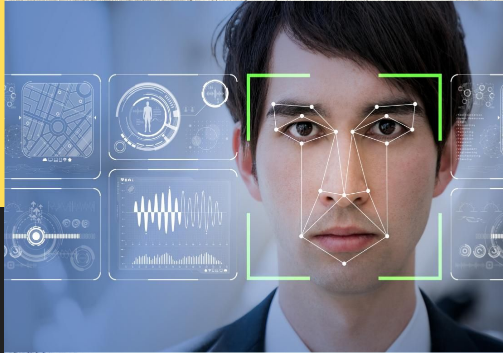
- Deep Learning: subset of machine learning focused on **neural nets**
    - Key components:
        - Neurons (nodes)
        - Layers
        - Weights and biases
        - Activation functions (ReLU, sigmoid, tanh)
        - Loss functions
        - Optimizers (SGD, Adam, RMSprop)
- Convolutional Neural Network (CNN)
    - A convolutional neural network is specifically designed to process grid-like data (i.e. images)
    - Layers:
        - Convolutional Layers: extract features from images
        - Activation Function: adds non-linearity to our network
        - Pooling Layers: reduces spatial dimensions
        - Fully Connected Layers: final classification/regression



Deep Learning in Computer Vision

- Deep Learning can have some very cool applications in computer vision
- Image Classification
  - Identifying objects, photos, or categories in images
- Object Detection
  - Locating and identifying multiple objects in a single image
- Face Recognition
  - Identifying and verifying individuals from facial features
- Image Generation
  - Creating new images from descriptions or other images
- Pose Estimation
  - Detecting human body positions and movements
- Style Transfer
  - Applying artistic styles to regular photographs

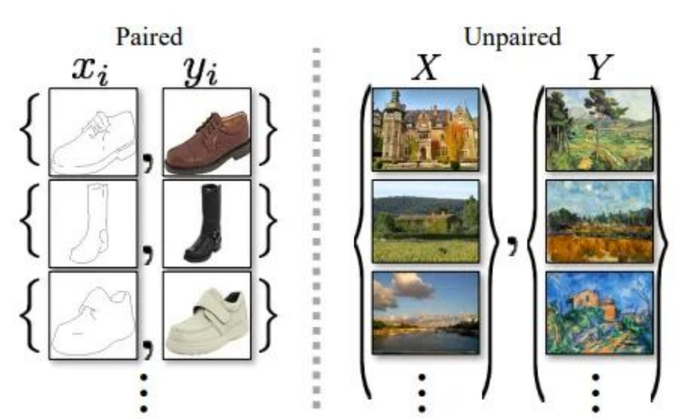




Today, we will be creating a style transfer model that applies a cartoon artistic style!

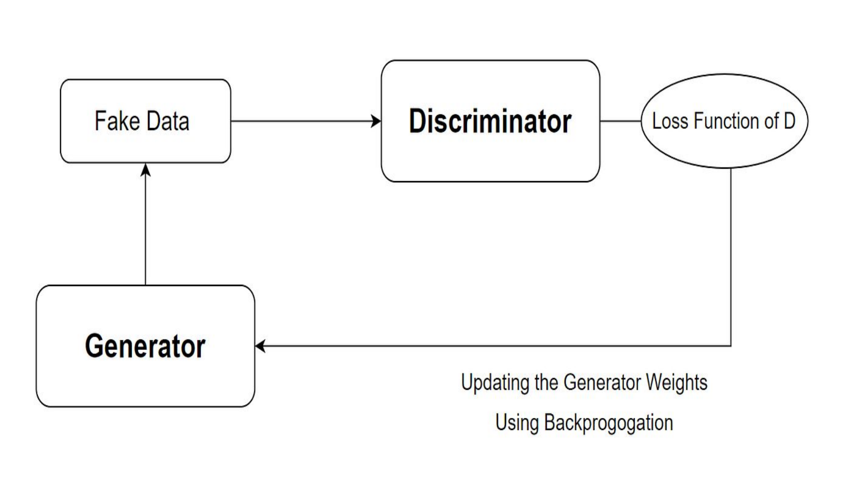
We will be creating a model for **style transfer!**

- **Generative Adversarial Network:** A deep learning model used often for image generation
  - A GAN has two components: a Generator and a Discriminator
  - The Generator learns to generate images (in this case, transform images to a cartoon style) and the Discriminator tries to “guess” whether the generated images are real or fake
- We will be implementing **CartoonGAN** – a GAN specifically optimized for cartoons!
  - Instead of just predicting between fake and real images, CartoonGAN tries to predict against a smoothed version of the cartoon images, which makes it generate better images
  - CartoonGAN works on **unpaired** images - images that do not correspond to one another



Model Training

- **Step 1: Train Discriminator**
  - Shows D three types of images:
    - Real cartoons (should say "real")
    - Smoothed cartoons (should say "fake")
    - Generated images (should say "fake")
  - Like training an art critic to spot different styles
- **Step 2: Train Generator**
  - G tries to create images that:
    - Look like cartoons (fool D)
    - Preserve original photo content
  - Like an artist learning to balance style and content
- **Validation & Checkpointing**
  - Regularly tests G's performance on new photos
  - Saves the best performing model
  - Like keeping your best artwork for reference



Model Inference

- At the end, our trained Generator can be used for generating realistic cartoon images!
- We will use our Generator specifically to convert images to cartoons