MGMT298D
Science and Strategy of AI

Week 5

Computer Vision and Convolutional Neural Networks

Auyon Siddiq
UCLA Anderson School of Management

Neural Networks Recap

The deep learning recipe:

Many training examples + stochastic gradient descent + GPUs

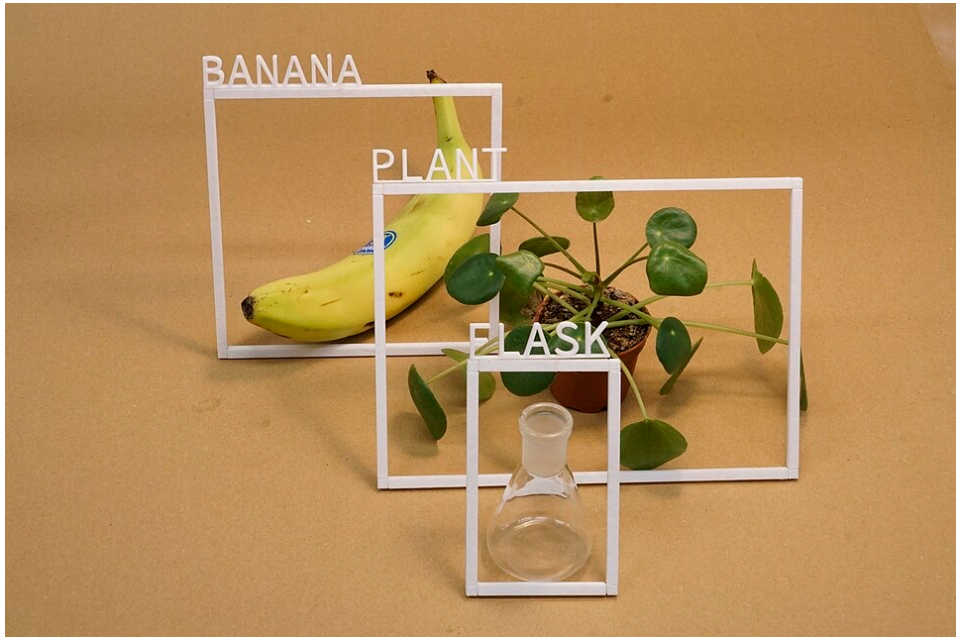
Training examples are the **data** the network learns from (features + true labels)

Stochastic gradient descent (aka backprop) is the **algorithm** that updates network weights iteratively

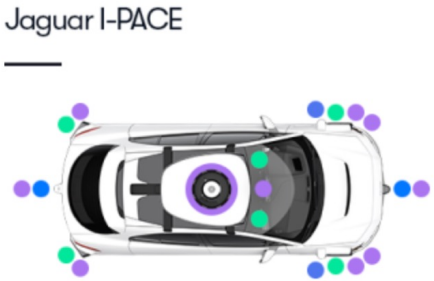
GPUs provide the enormous **compute** required to execute the large number of multiplication/additions inherent in SGD / backprop

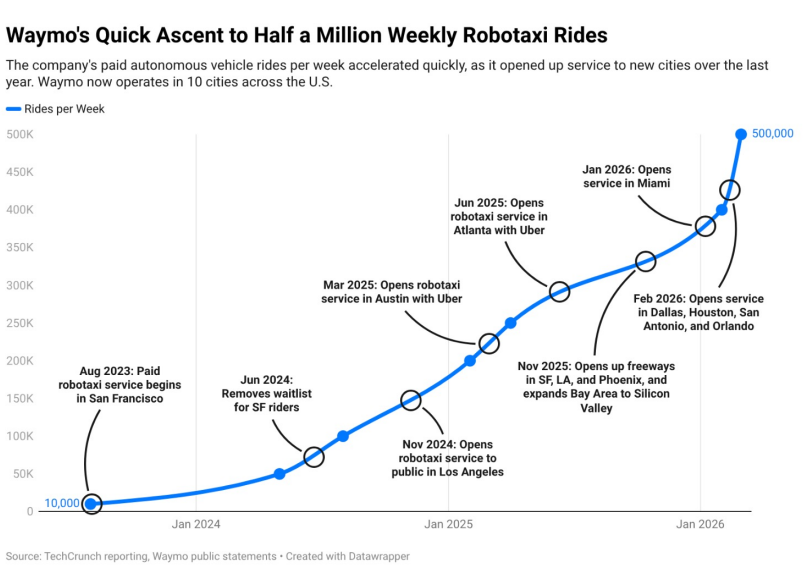
Computer Vision

Computer vision: branch of AI focused on interpreting images

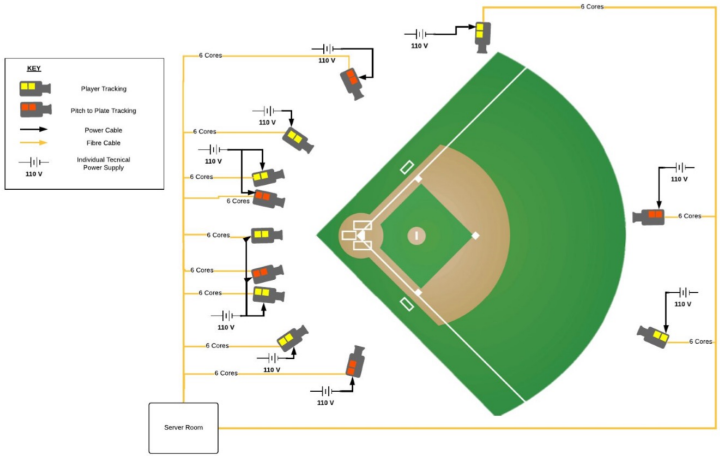


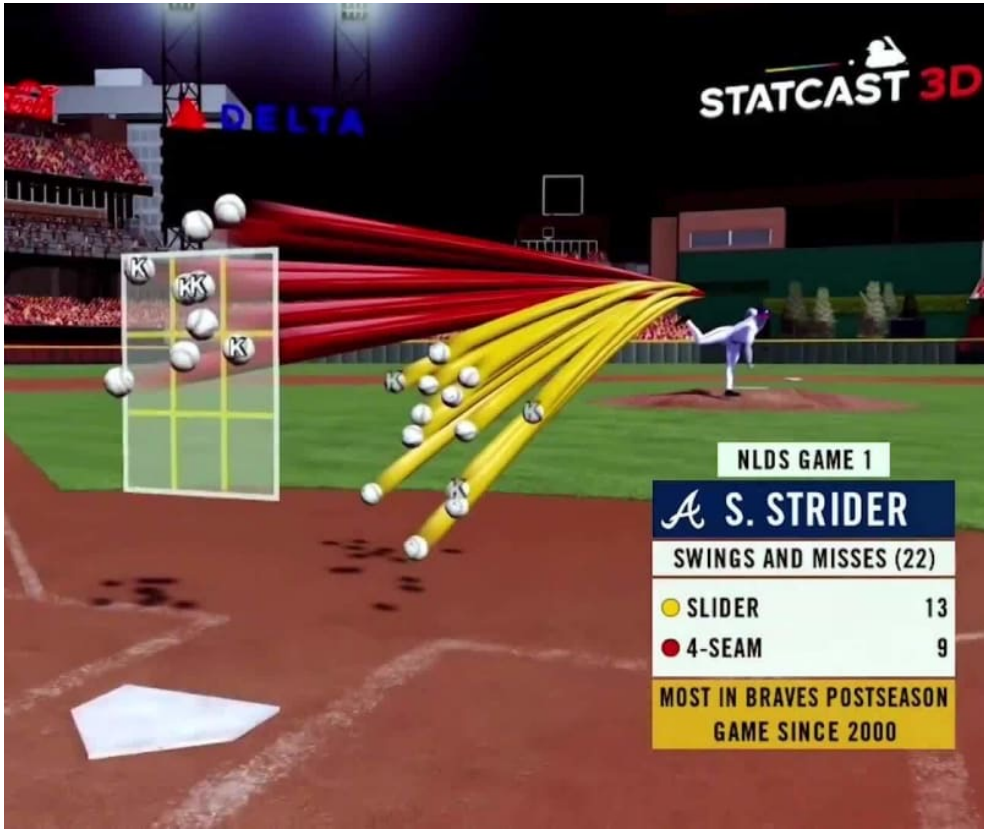
Autonomous Vehicles



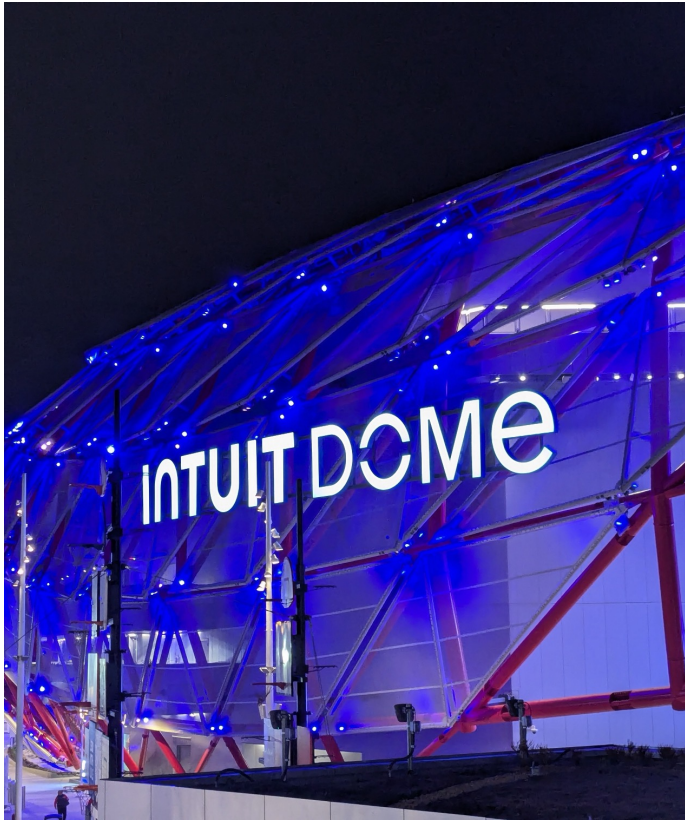


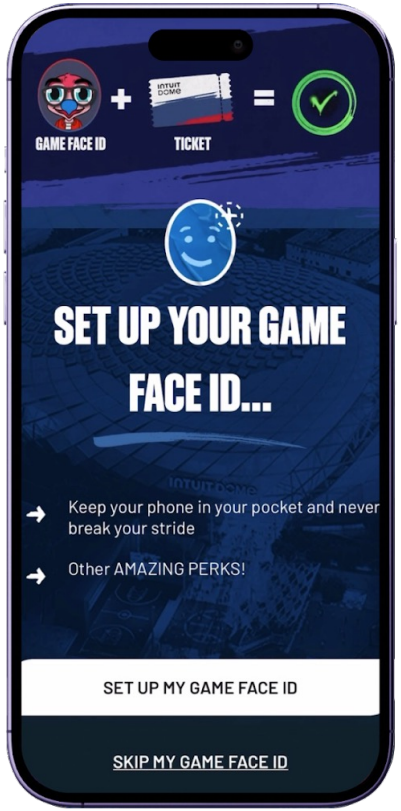
Hawk-Eye


KEY
Player Tracking
Pitch to Plate Tracking
Power Cable
Fibre Cable
Individual Tecnical Power Supply
110 V
6 Cores
Server Room


DELTA
STATCAST 3D
NLDS GAME 1
S. STRIDER
SWINGS AND MISSES (22)
SLIDER 13
4-SEAM 9
MOST IN BRAVES POSTSEASON GAME SINCE 2000

Game Face ID at Intuit Dome





How many cameras...?

Ralph Lauren

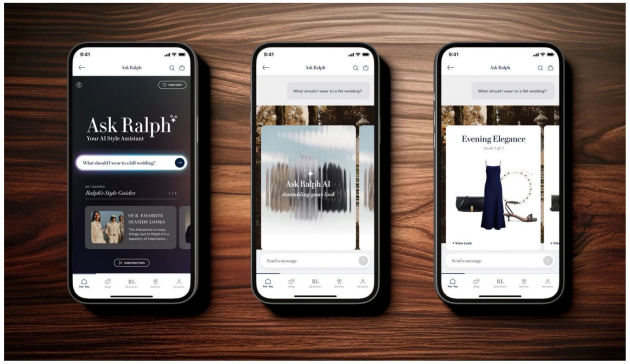
TECHNOLOGY

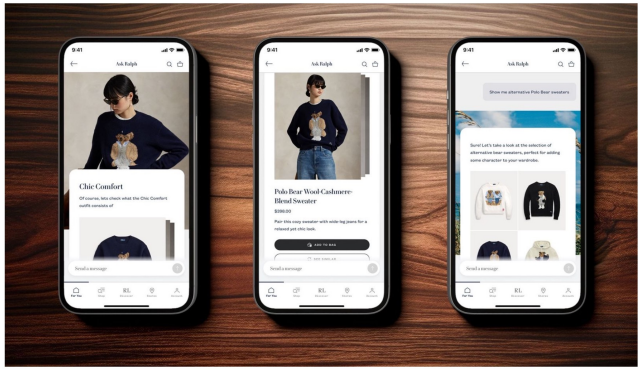
Inside Ralph Lauren's new white-label AI styling tool

Chief branding and innovation officer David Lauren is betting on generative AI to improve personalisation and customer engagement at the heritage brand.

September 9, 2025

(Article posted on BruinLearn)

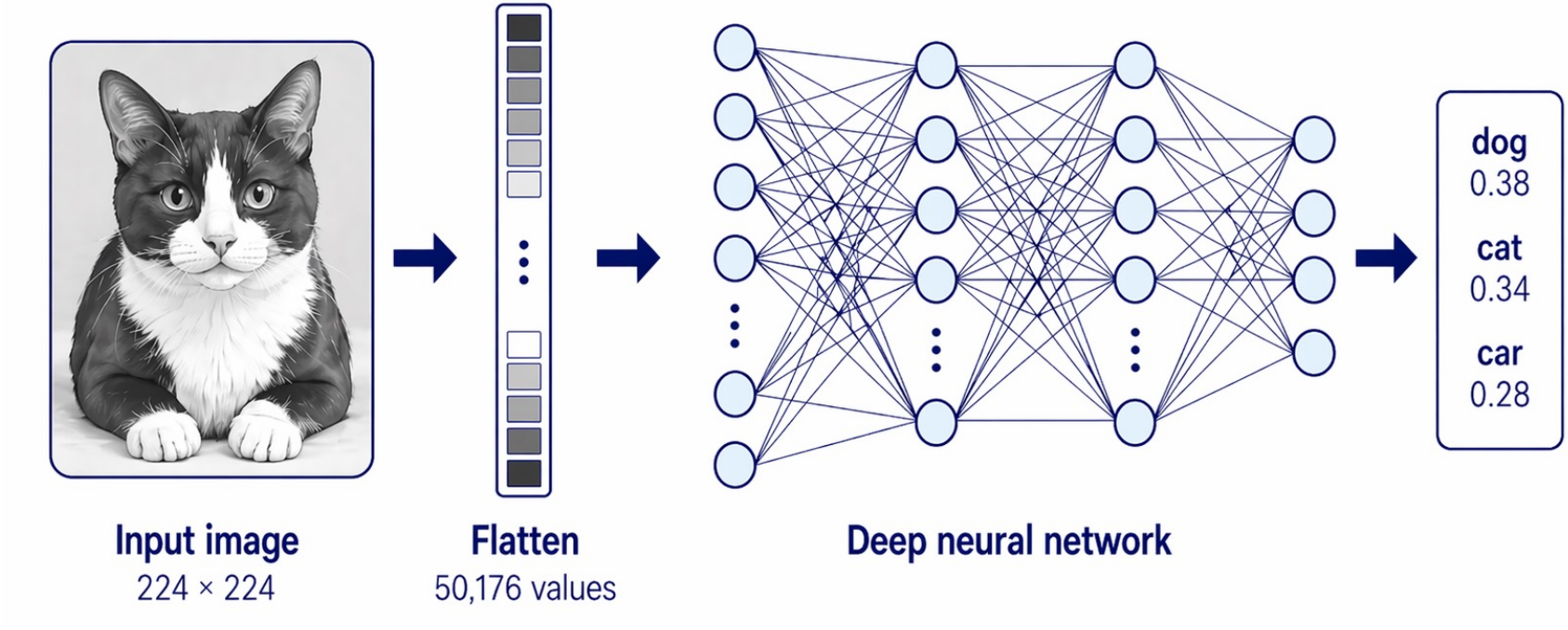




The Ask Ralph tool will refine its recommendations to more detailed user promptsPhoto: Courtesy of Ralph Lauren

John Deere See and Spray

Why not use deep NNs?



Today's Class: Computer Vision

How do neural networks work on images?

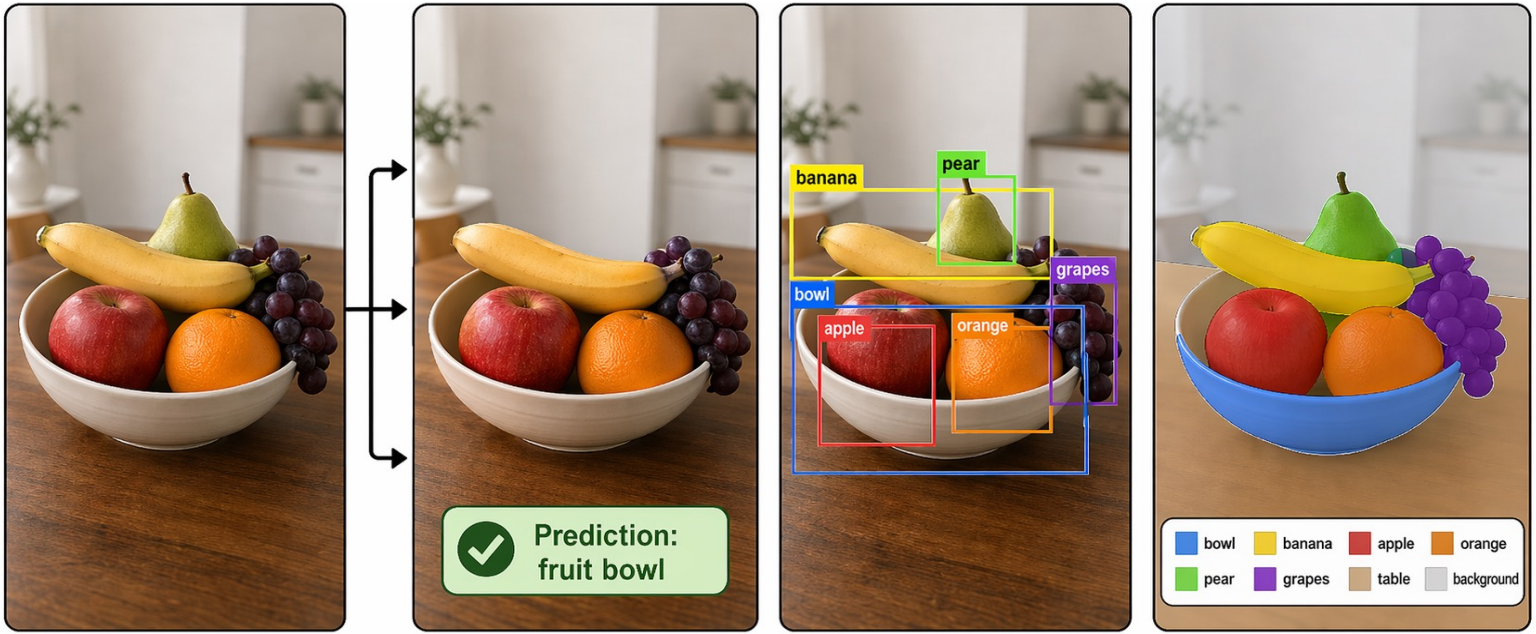
Convolutional neural networks (CNNs) are the sub-field of deep learning focused on this

We will cover the building blocks for the standard CNN architecture, and cover some best practices for deploying CNNs

Questions for deployment:

- Should we train a model from scratch?
- Can we use a pre-trained model?
- What other options are there?
- What are the data requirements?

Computer Vision: Three Task Types



Input image

Image classification
1 class for whole image

Object detection
Find and classify
multiple objects

Segmentation
Classify every pixel

Convolution Filters

Convolutional Filters

Key idea: a convolutional filter is a small window of numbers, typically 3×3 or 5×5

Convolutional filters detect shapes: horizontal lines, vertical lines, etc.

detects horizontal lines

1	1	1
0	0	0
-1	-1	-1

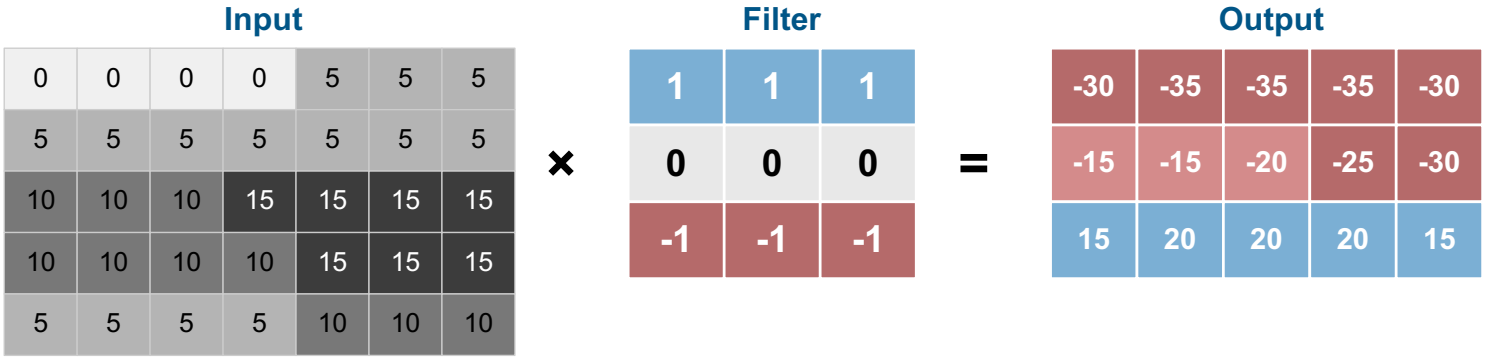
detects vertical lines

1	0	-1
1	0	-1
1	0	-1

Blue = positive (1): looks for brightness **Red = negative (−1):** looks for darkness **Gray = zero:** ignores

Convolutional Filter: Example

Slide the filter across the image, multiply and sum at each position:



After ReLU: What the Network "Sees"

ReLU clamps all negative values to zero — only positive activations survive

Raw Convolution Output

-30	-35	-35	-35	-30
-15	-15	-20	-25	-30
15	20	20	20	15

→ **ReLU** →

After ReLU

0	0	0	0	0
0	0	0	0	0
15	20	20	20	15

The bottom row lights up — that's where the horizontal edge is.

Each value is a score for "how strongly does this location contain a horizontal edge."

The output is a heat map of horizontal edges in the original image.

Convolutional Layer

A group of filters creates a convolutional **layer**

Each filter is a specialist — one detects horizontal lines, another detects vertical lines, another detects curves, etc.

The network learns which filters are useful through training

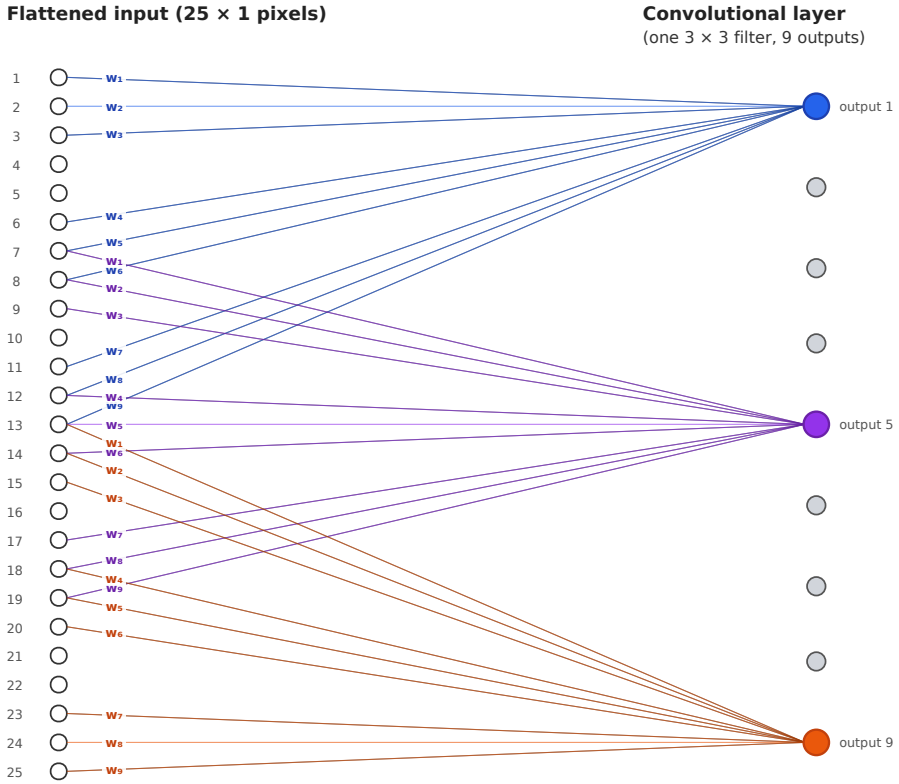
Excel example

How are convolutional filters designed?

Previously designed by hand, now treated as weights that can be learned

Example: 5x5 image with 3x3 filter

9 different ways to place a 3x3 filter on a 5x5 image



Original 5 × 5 image

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

- top-left patch → output 1
- center patch → output 5
- bottom-right patch → output 9

Shared 3 × 3 filter

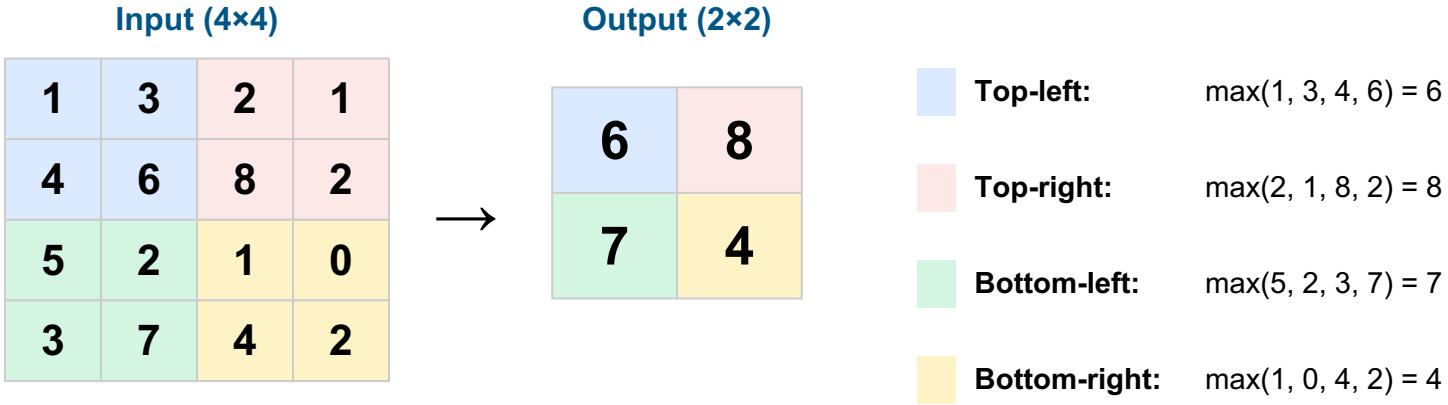
w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Pooling: Shrinking for Efficiency

Pooling summarizes small regions → leads to smaller network / fewer weights

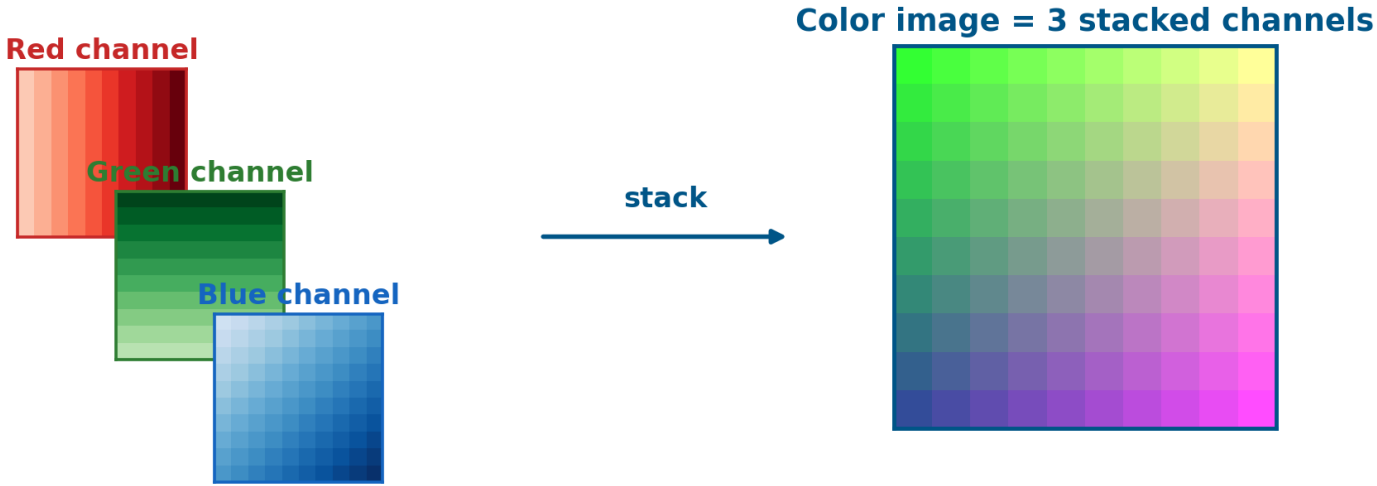
Max pooling: take the largest value in each 2×2 region

A 224×224 feature map becomes 112×112 — a quarter of the pixels



Color Images

Color images have 3 channels, e.g., one image is 224 x 224 x 3 pixels



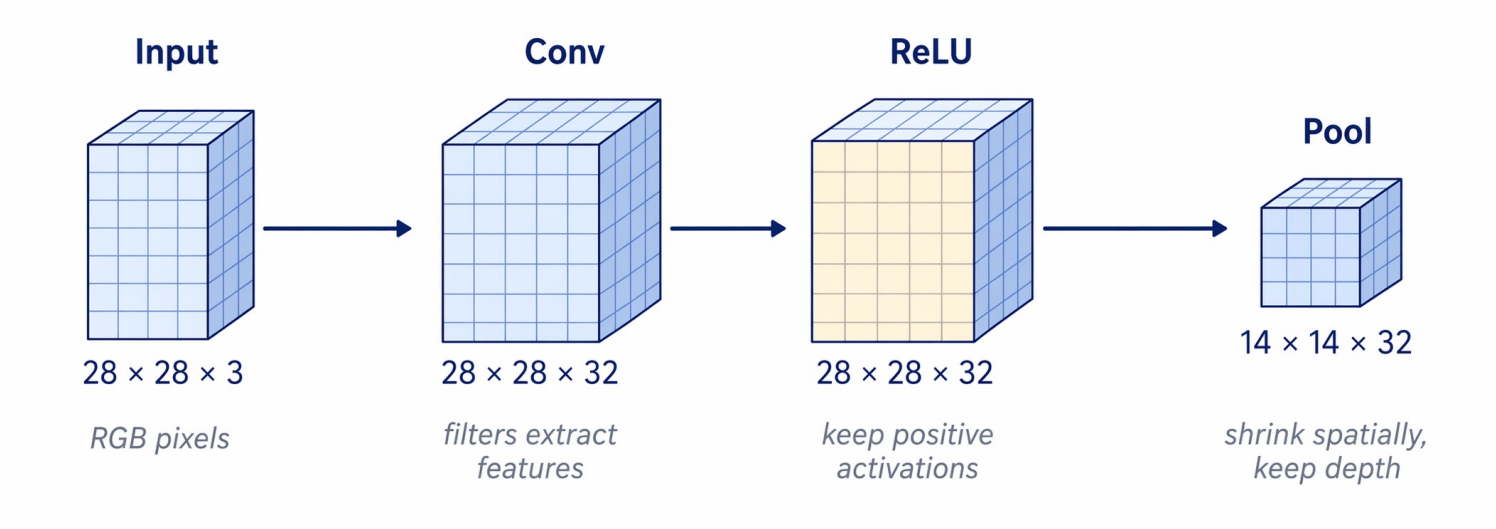
Each convolution filter is 3 x 3 x 3 – output is based on sum across all 27 weights

CNN Architecture

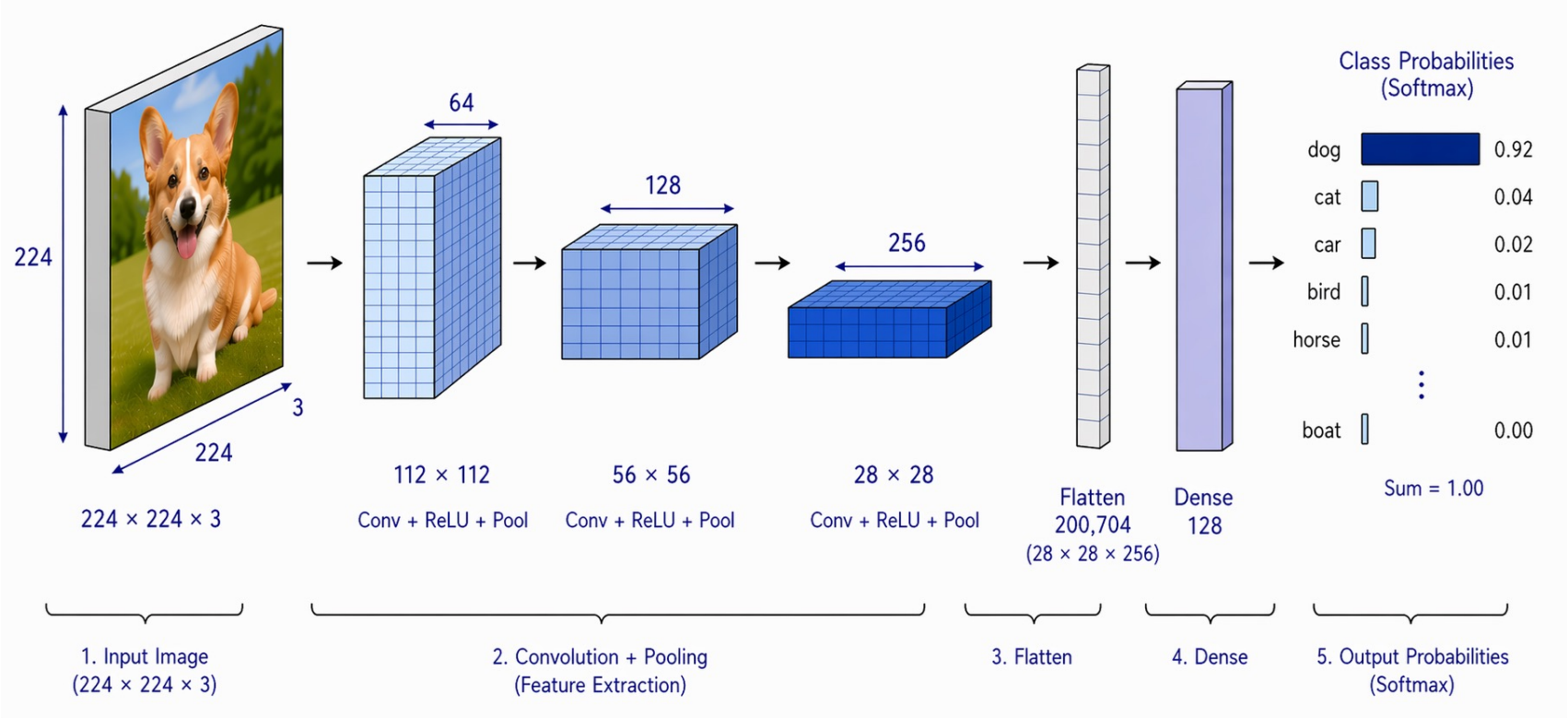
A Convolutional Block

A standard building block: apply filters, pass through ReLU, then downsample

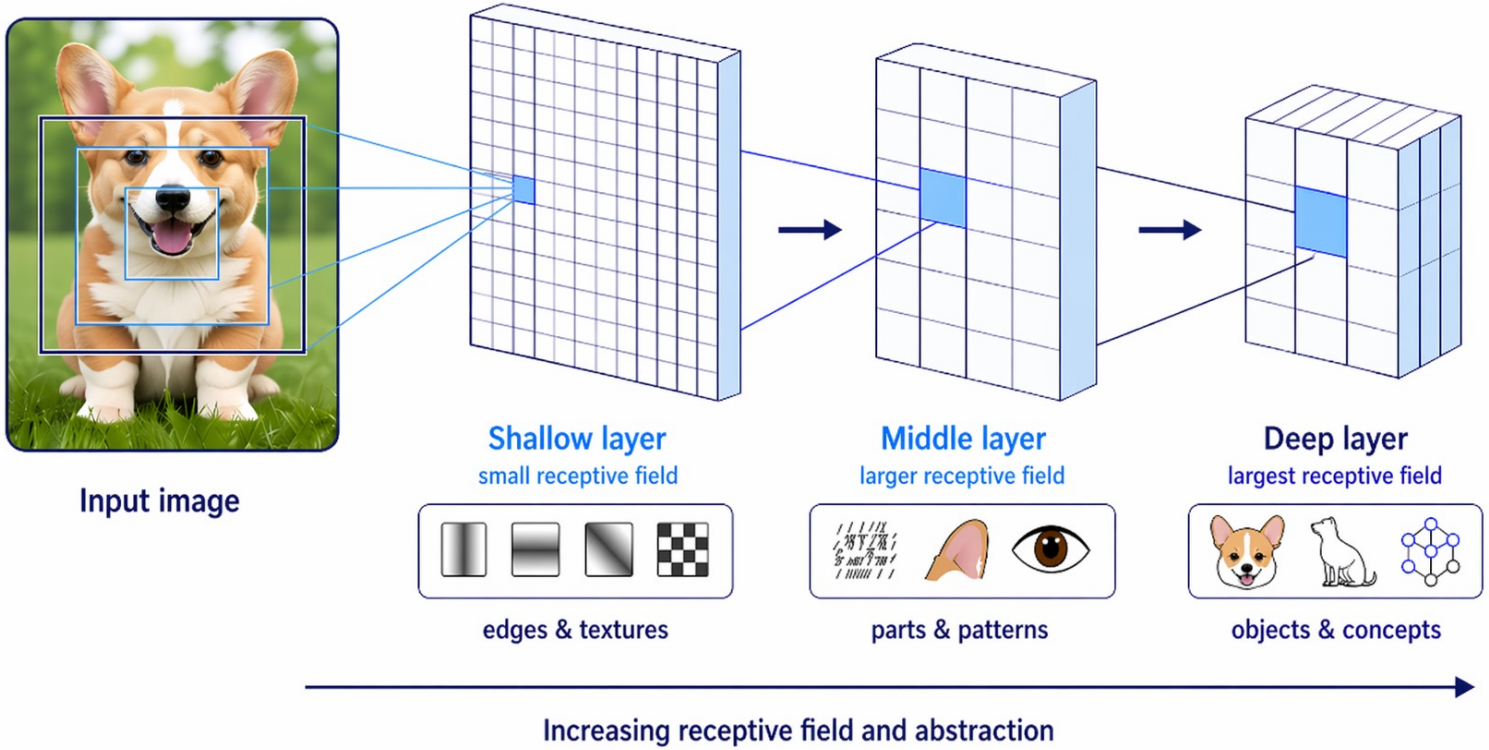
Can stack multiple blocks in a network to detect more complex features



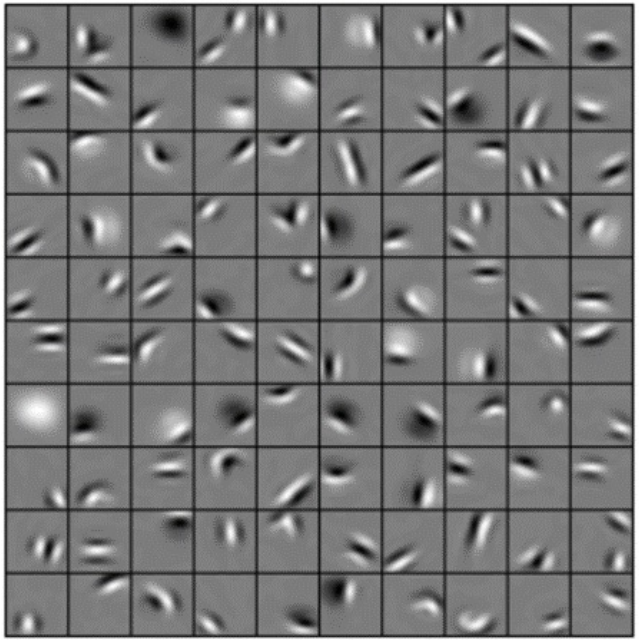
CNN Architecture

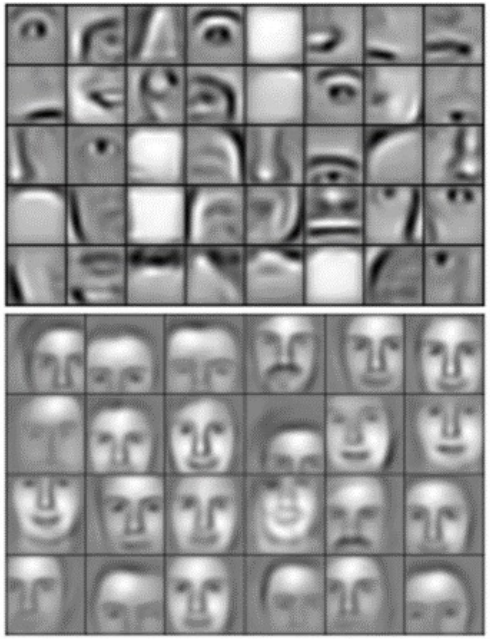


What Do Filters Learn?



What Do Filters Learn?





"What kind of input makes each filter activate?"

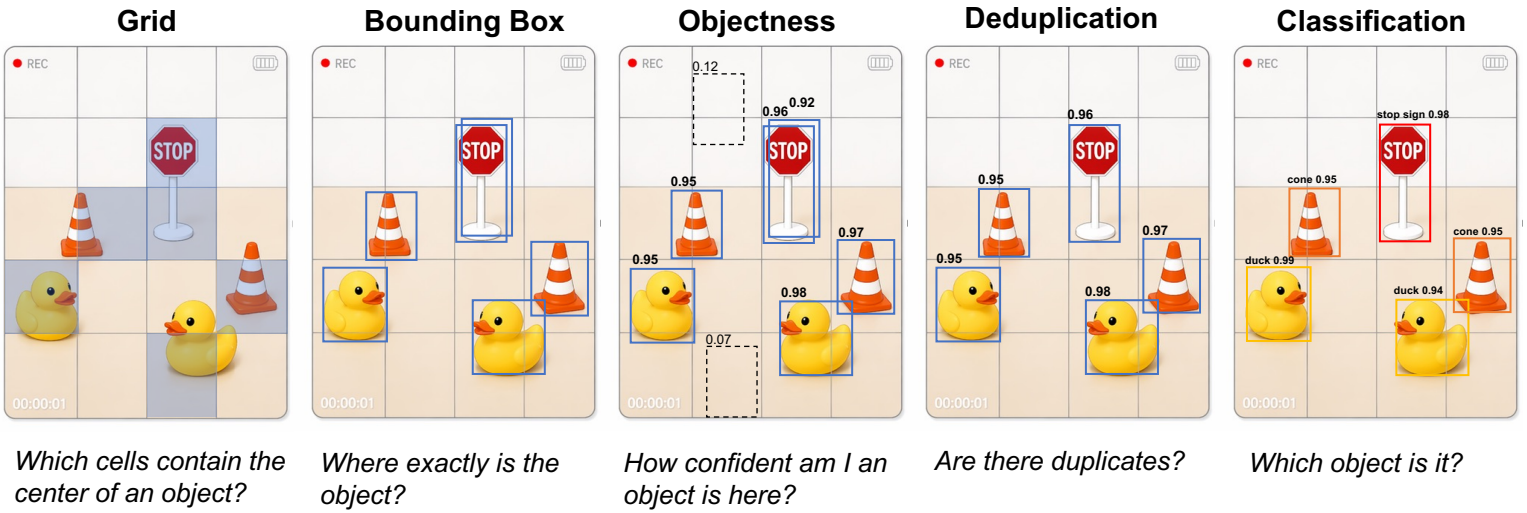
https://web.eecs.umich.edu/~honglak/icml09-ConvolutionalDeepBeliefNetworks.pdf


colab

15-min Break

Real-Time Object Detection

Divide the image into a grid; each cell predicts bounding boxes and class probabilities at once



Which cells contain the center of an object?

Where exactly is the object?

How confident am I an object is here?

Are there duplicates?

Which object is it?

DuckieBot is based on the ”YOLO” (You Only Look Once) model, which is industry-strength for real-time detection (see e.g., https://www.ultralytics.com/customers)


DUCKIETOWN

Deploying Computer Vision Models in Practice

Questions for Deployment

Should we train a model from scratch?

Can we use a standard model “off-the-shelf”?

Is there a hybrid approach?

What are the data requirements?

Option 1: Use a Fully Pre-Trained Model

Why re-invent the wheel? Many applications built on a few well-known models

Good for fast and approximate visual clustering and tagging; zero data or training required

Model	Architecture	Params	Use cases
ResNet-50	50-layer deep CNN; ~16 convolutional blocks	~25M	Default vision backbone (e.g., used by FB Marketplace, Bing Image Search)
MobileNetV2	53-layer lightweight CNN; 17 convolutional blocks	~3.4M	On-device mobile vision; powers Google's ML Kit (25,000+ apps); augmented reality; image labeling
YOLO	CNN with 24 convolutional blocks, plus detection head for bounding boxes	~7.5M	Real-time detection; robotics, agriculture, surveillance, manufacturing; traffic cameras

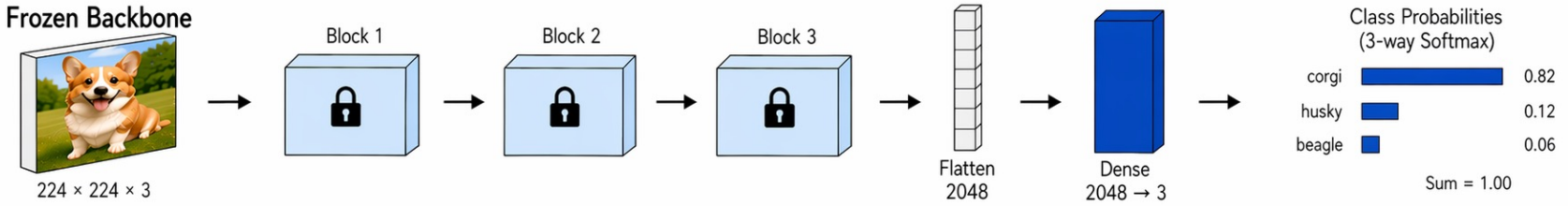
Most useful when not doing strict classification, e.g. High-level clustering of product images, finding visually similar products

Option 2: Transfer Learning (Frozen Backbone)

Model supplies the representation of the image, you decide what to do with it

E.g., ResNet-50 defaults to 1000 classes, replace final classification head with your own (corgi, husky, beagle)

Very light on training and data (100s of labeled example images)



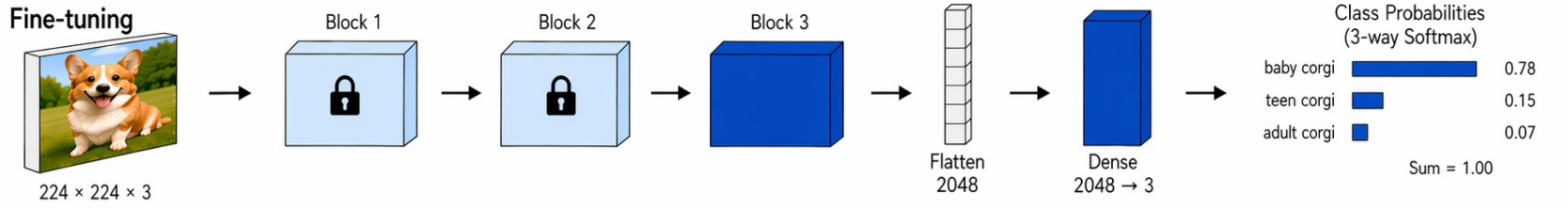
Useful for domain-specific classification tasks when you have specific high-level products to tag, e.g., shoes vs tops vs outerwear

Option 3: Transfer Learning (Fine-Tune Backbone)

In addition to custom classification head, update last convolutional layers

You decide how deep into the backbone to train the model

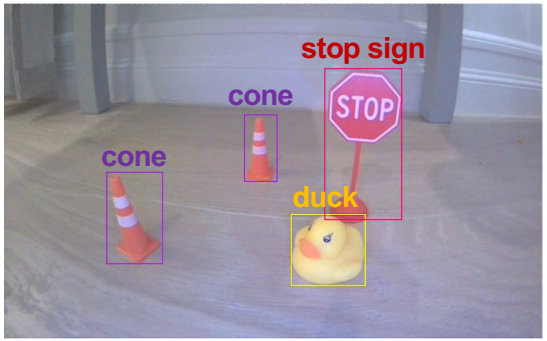
More demanding on training and data (1,000-10,000 labeled example images)



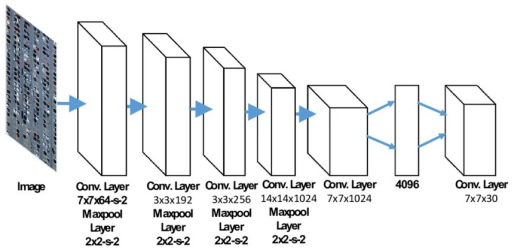
Useful for subtle classification tasks that are highly domain-specific, e.g., distinguish between stain vs. tear on returned apparel

Fine-Tuning CNN for DuckieBot

Collect new training images

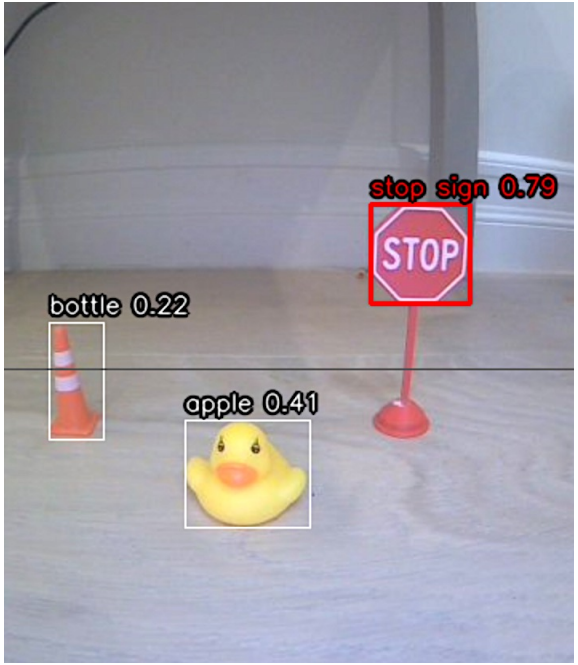


Label with bounding boxes and class

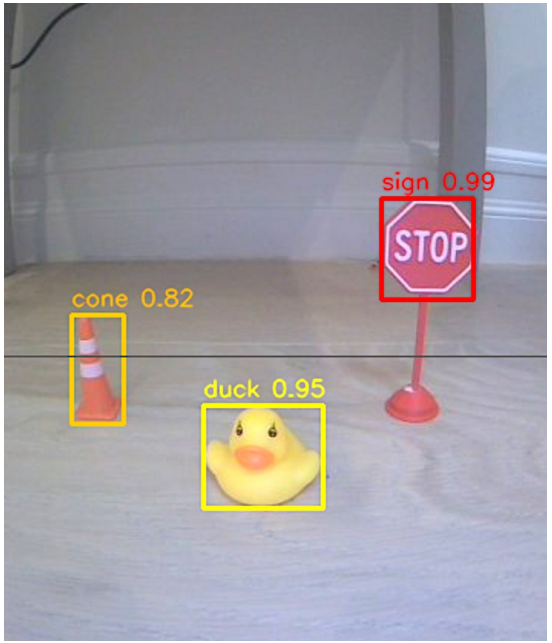


Update weights of pre-trained CNN

Fine-Tuning CNN for DuckieBot



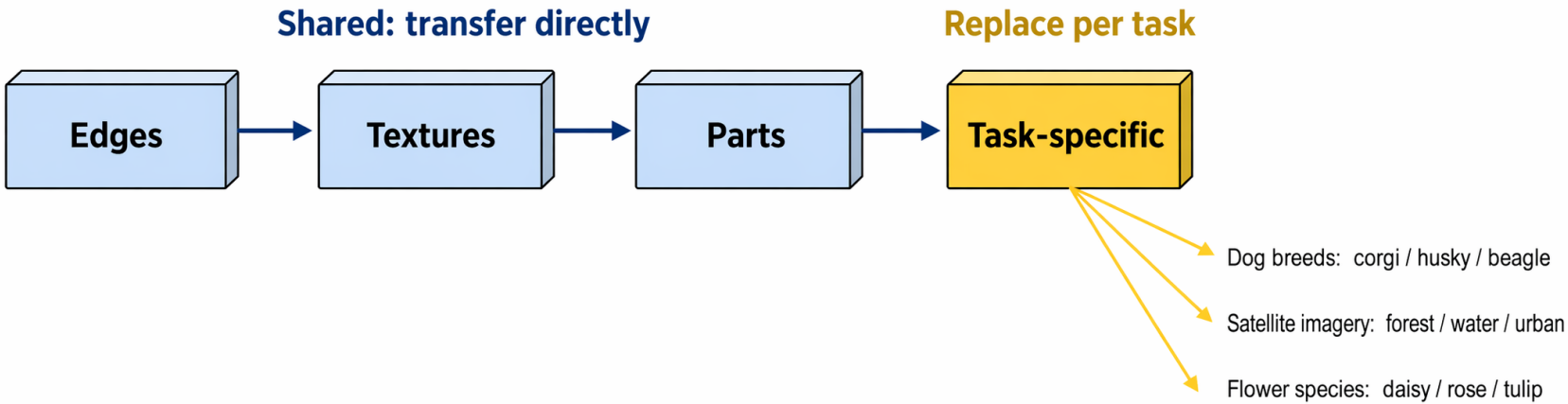
Without fine-tuning
(pre-trained CNN)



Fine tuned on 300 images

Transfer Learning: Intuition

Why does transfer learning work?



Visual signals like edges, textures and parts are shared across nearly every image task

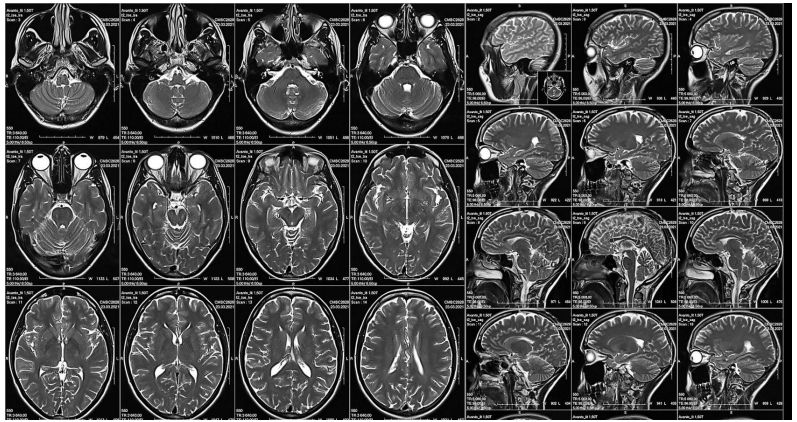
Option 4: Train CNN from Scratch

Train the whole CNN – you decide architecture, all parameters, optimization, etc.

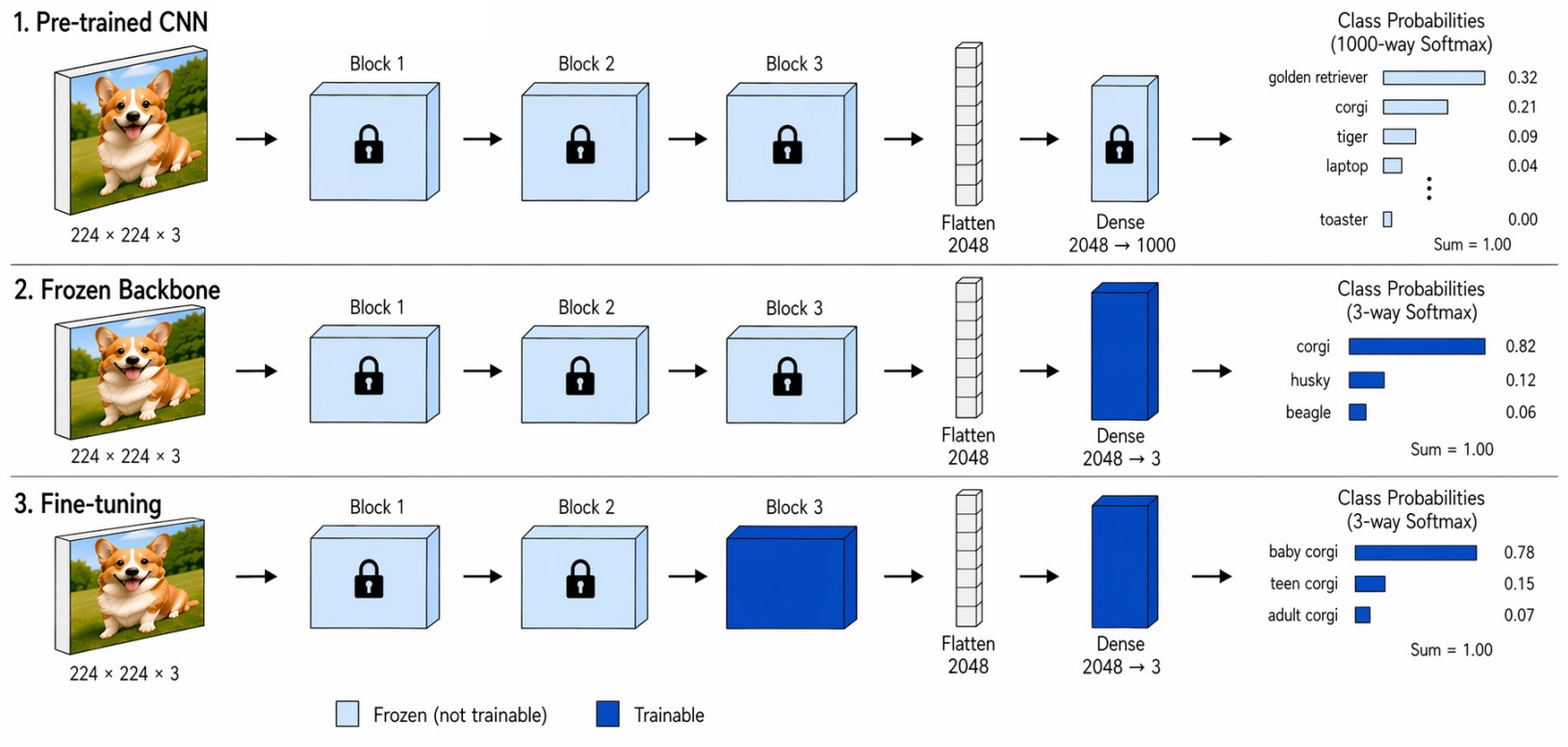
Very demanding on training and data (ideally 100,000+ labeled example images), need to buy compute, probably in-house AI team

Most likely overkill in practice for most applications unless dealing with highly specialized classification / detection tasks

Most applicable for detecting non-standard visual patterns that existing models are unlikely to recognize, e.g., medical imaging



Transfer Learning Summary



CNN Deployment Approaches

	Pre-trained	Frozen backbone	Fine-tuning	Train from scratch
Data required	0	100-1,000	1,000-100,000	100,000+
Training cost / time	None	minutes to hours, single GPU	hours to a day on a GPU	days to weeks, multiple GPUs
Use cases	Visual similarity, clustering, quick tagging	Custom labels on common images	Capturing subtle distinctions (e.g., fine-grained defects, style)	Specialized domains; proprietary dataset, integration with other AI

Increasing data + compute requirements

Best Practices: Data Augmentation

Expand the training set with modified copies of each image – get new data “for free"

Flip, rotate, crop, resize, color jitter

original

still a boot

still a boot

still a boot

still boots


colab

Assignment 5

Glossary (1/2)

CNN	A neural network for images that uses small sliding filters to detect spatial patterns like edges and shapes.
Convolutional Filter	A small grid of learnable weights (e.g., 3×3) that slides across an image to detect a specific pattern.
Pooling	A downsampling operation (typically max pooling) that shrinks feature maps, keeping the strongest signals.
Transfer Learning	Reusing a model pre-trained on a large dataset (like ImageNet) as the starting point for a new task.
Frozen Backbone	Keeping the pre-trained layers of a model fixed during training, so only the new classification head learns.

Glossary (2/2)

Fine-Tuning	Unfreezing some or all layers of a pre-trained model and continuing training on your own dataset.
ImageNet	A benchmark dataset of 1.2 million labeled images across 1,000 categories, widely used for pre-training.
Object Detection	Identifying what objects are in an image and where they are located, using bounding boxes and class labels.
ResNet	A widely used CNN architecture for training very deep networks (50–152 layers).