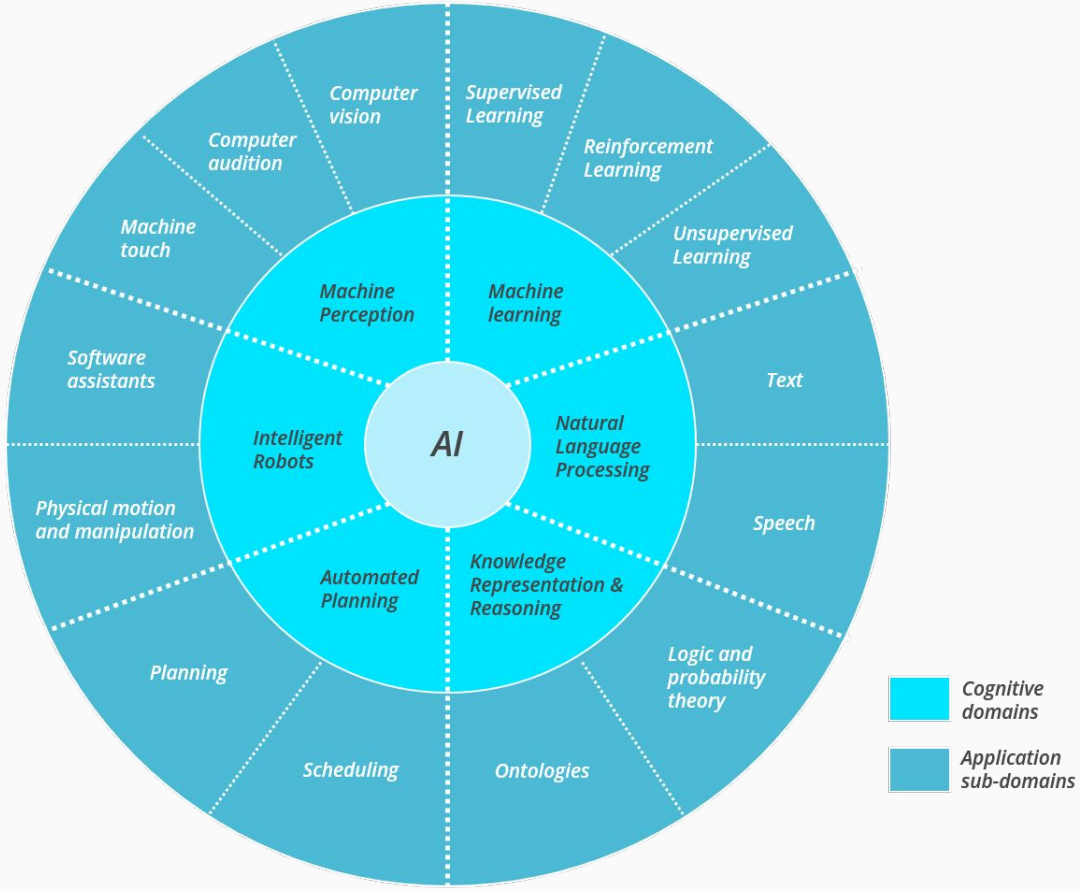
Introduction to Machine Learning and Artificial Intelligence

SCALER EDGE

Your Instructors

1. **Abhimanyu Saxena**
   Co founder, Scaler Academy
2. **Naman Bhalla**
   Incoming Software Engineer, **Google**. Instructor at Scaler Edge.
3. **Tarun Malhotra**
   Ex Machine learning engineer at **Media.net**. Lead instructor at Scaler Edge.

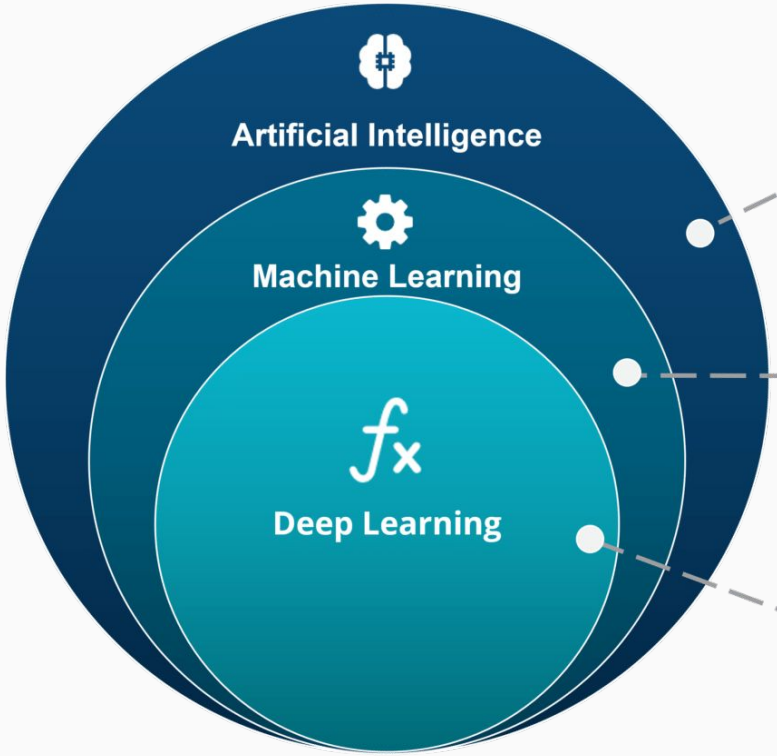
What will we cover?


AI
Machine Perception
Machine learning
Intelligent Robots
Natural Language Processing
Automated Planning
Knowledge Representation & Reasoning
Computer vision
Supervised Learning
Reinforcement Learning
Unsupervised Learning
Text
Speech
Logic and probability theory
Ontologies
Scheduling
Planning
Physical motion and manipulation
Software assistants
Machine touch
Computer audition
Cognitive domains
Application sub-domains

AI ! What is It?

Simply, It is making artificial things, like Computers, INTELLIGENT!

But HOW?



ARTIFICIAL INTELLIGENCE

A technique which enables machines to mimic human behaviour

MACHINE LEARNING

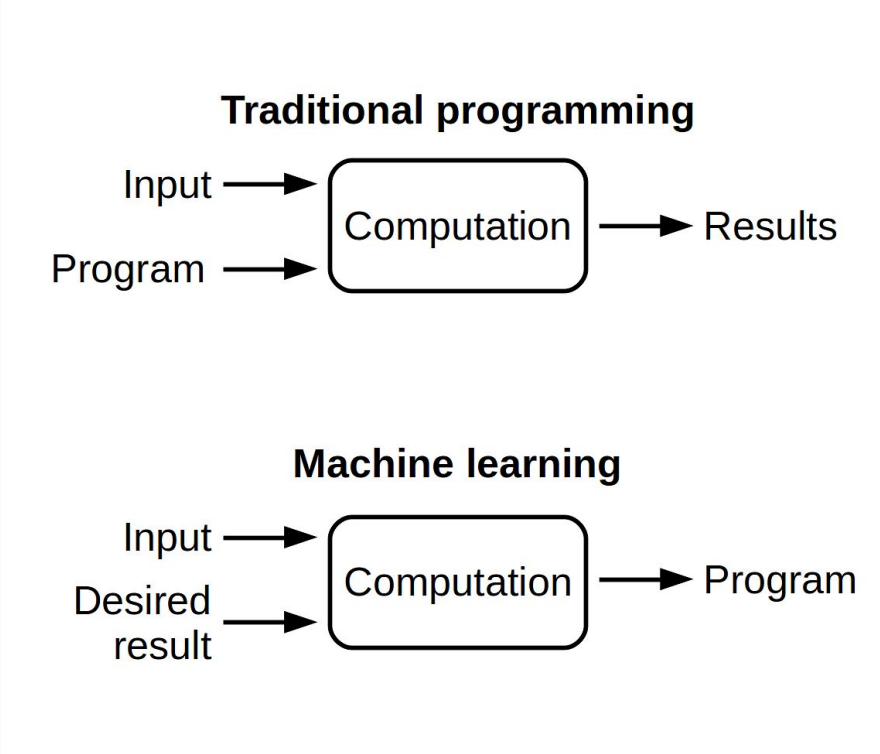
Subset of AI technique which use statistical methods to enable machines to improve with experience

DEEP LEARNING

Subset of ML which make the computation of multi-layer neural network feasible

Machine Learning!

Ability of computer to learn without being explicitly programmed.


Traditional programming
Input
Program
Computation
Results
Machine learning
Input
Desired result
Computation
Program

How to Make ML Happen?

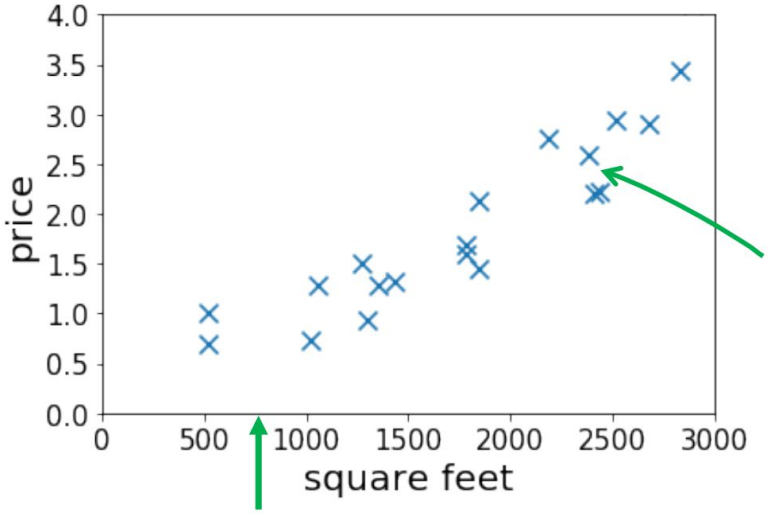
Supervised Learning

Supervised Learning

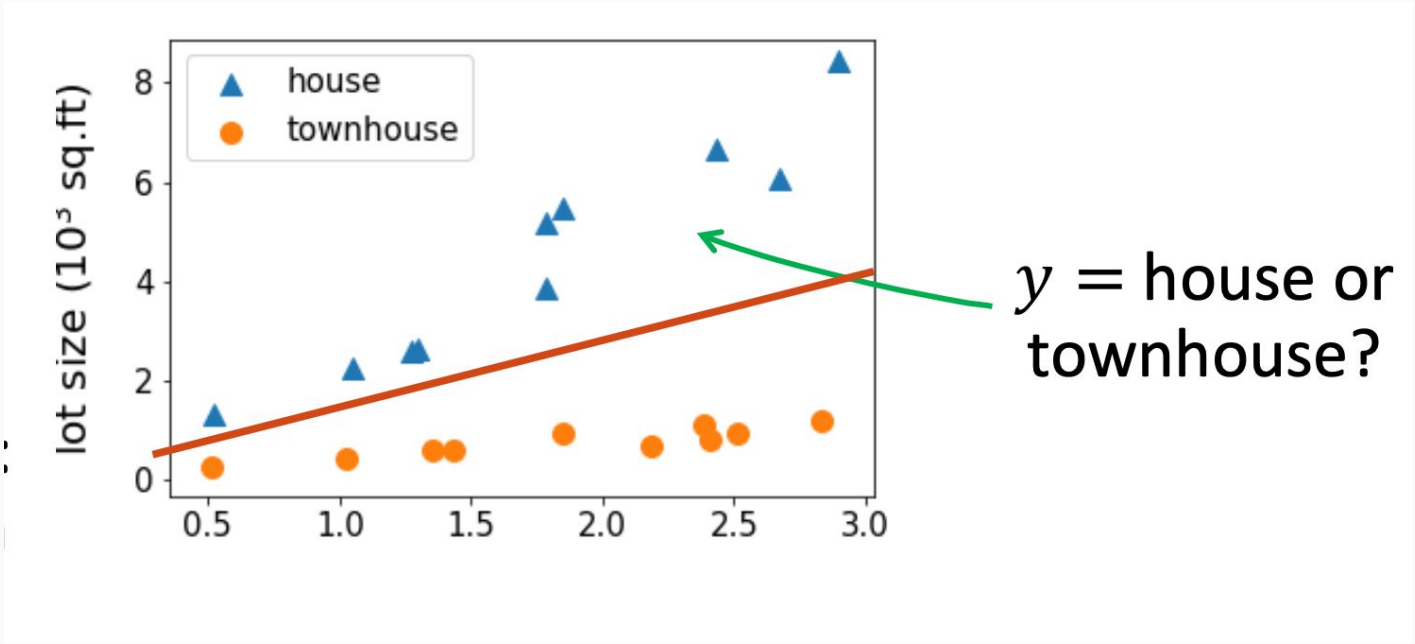
- You give input-output pairs!

“Hey Mr. Computer! This is a Cat. This is also a Cat. And here, it is a Dog. And here are a few more dogs for you. And ofcourse, a few more Cats. Today I have told you what they are. Now learn to figure them out yourself for future!”

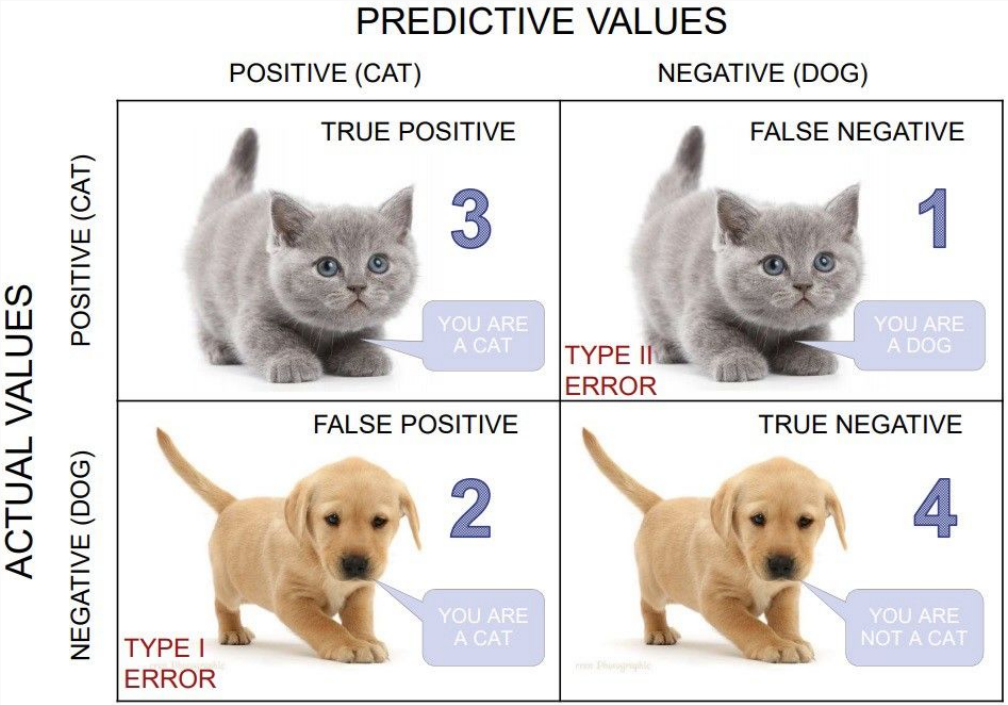
House Pricing Prediction



House Type Prediction



And of course, Are you a Cat?



SCALER EDGE

Unsupervised Learning

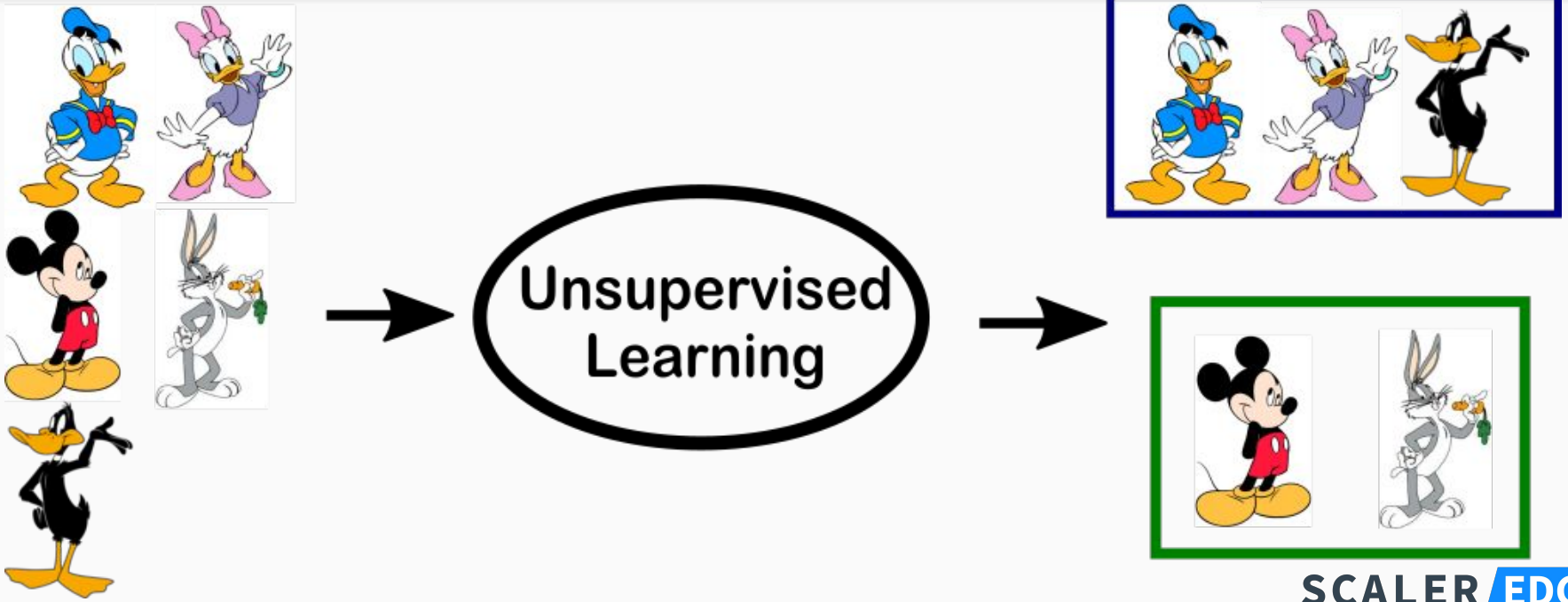
SCALER EDGE

Unsupervised Learning

- Only input Data. No labeled responses.

“Hey Mr. Computer! See. There is a lot of data here. I don’t know what is going with it. Can you help me find some cool patterns that exist?”

Unsupervised Learning

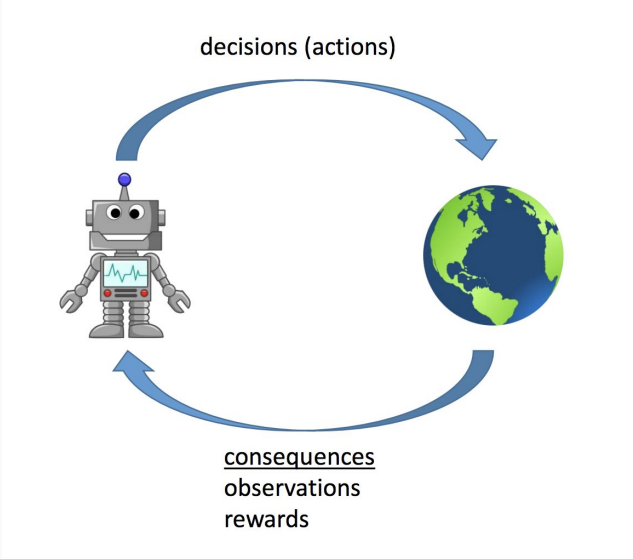


Reinforcement Learning

SCALER EDGE

Reinforcement Learning

- Tell the system to do actions that maximize a Reward!

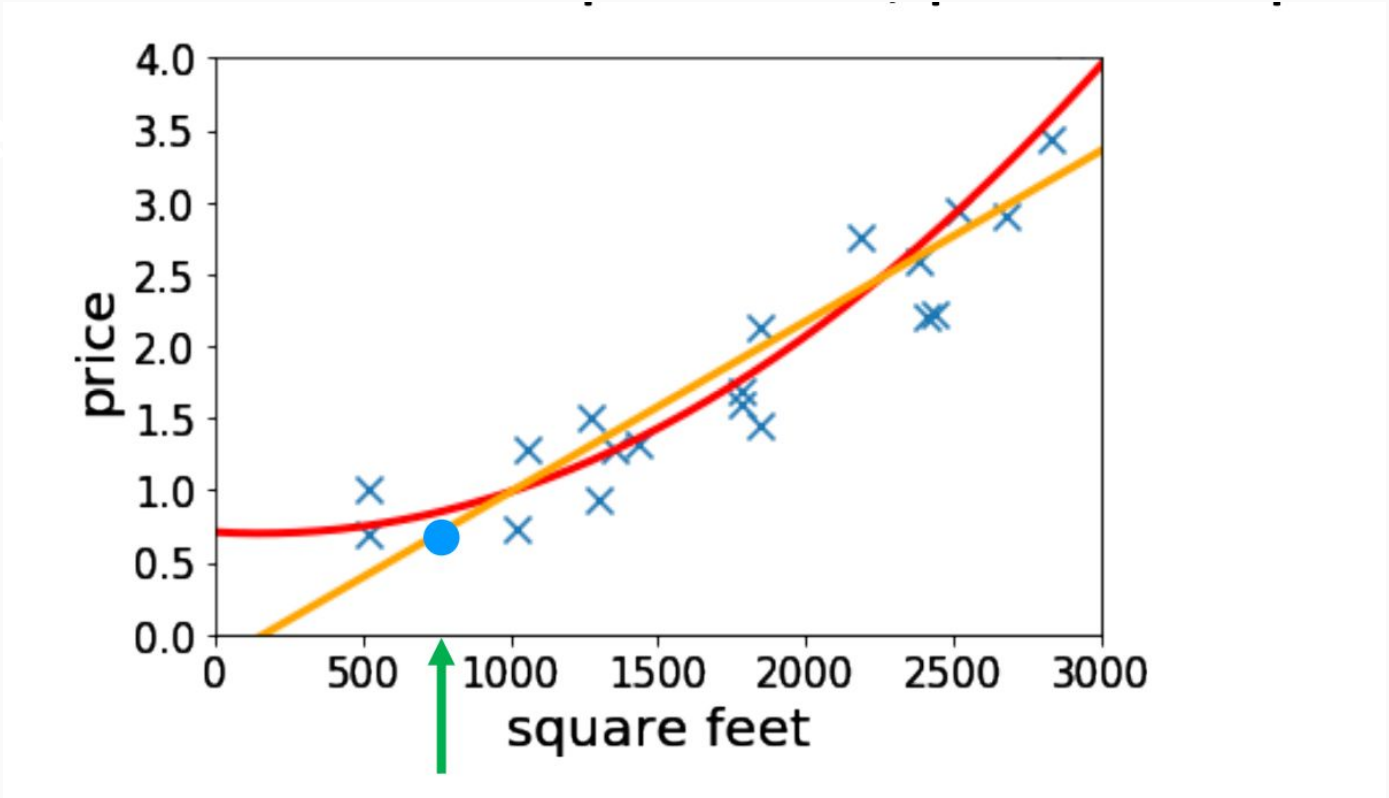


Everything is a Probability!

- Are you really sure you are attending this workshop?
- You might be sleeping actually and this is just a dream you are having?
- Watch The Matrix!
- You choose what looks most probable to you.

Linear Regression - The one where it all Begins (Generally!)

SCALER EDGE


4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
price
0
500
1000
1500
2000
2500
3000
square feet

Problem?

- I somehow need to know the function that is followed by house prices.
- Issues:
  - Humans are weird.
  - No function can map it exactly.
- Simplest Choice?
  - Linear function!!

Solution!

- How will I find this line?
  - Equation of Line? Y = mx + c
  - What is the x?
    - Independent Variable => House Size, etc.
  - Let’s try to guess initial values of m and c and create a line.
  - Now we find how far away were we from correct?
  - Let’s try to reduce this distance.
    - Distance?
    - Loss function!

Loss Function

- How well did the model predict upon the given data.
- How much deviation from the correct answer was there.
- Larger Values => Larger Loss => Bad model :(

Mean Absolute Error (L1 Loss Function)

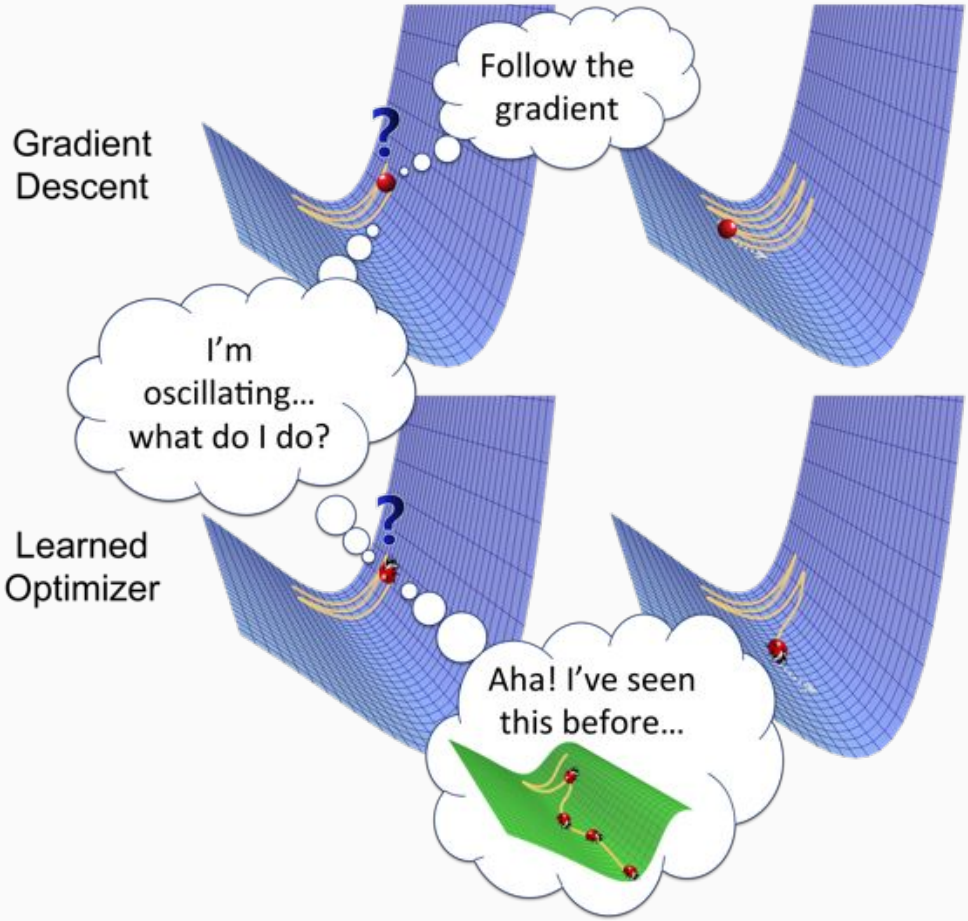
$$MAE = \frac{\sum_{i=1}^{n} | y_i - \hat{y}_i |}{n}$$

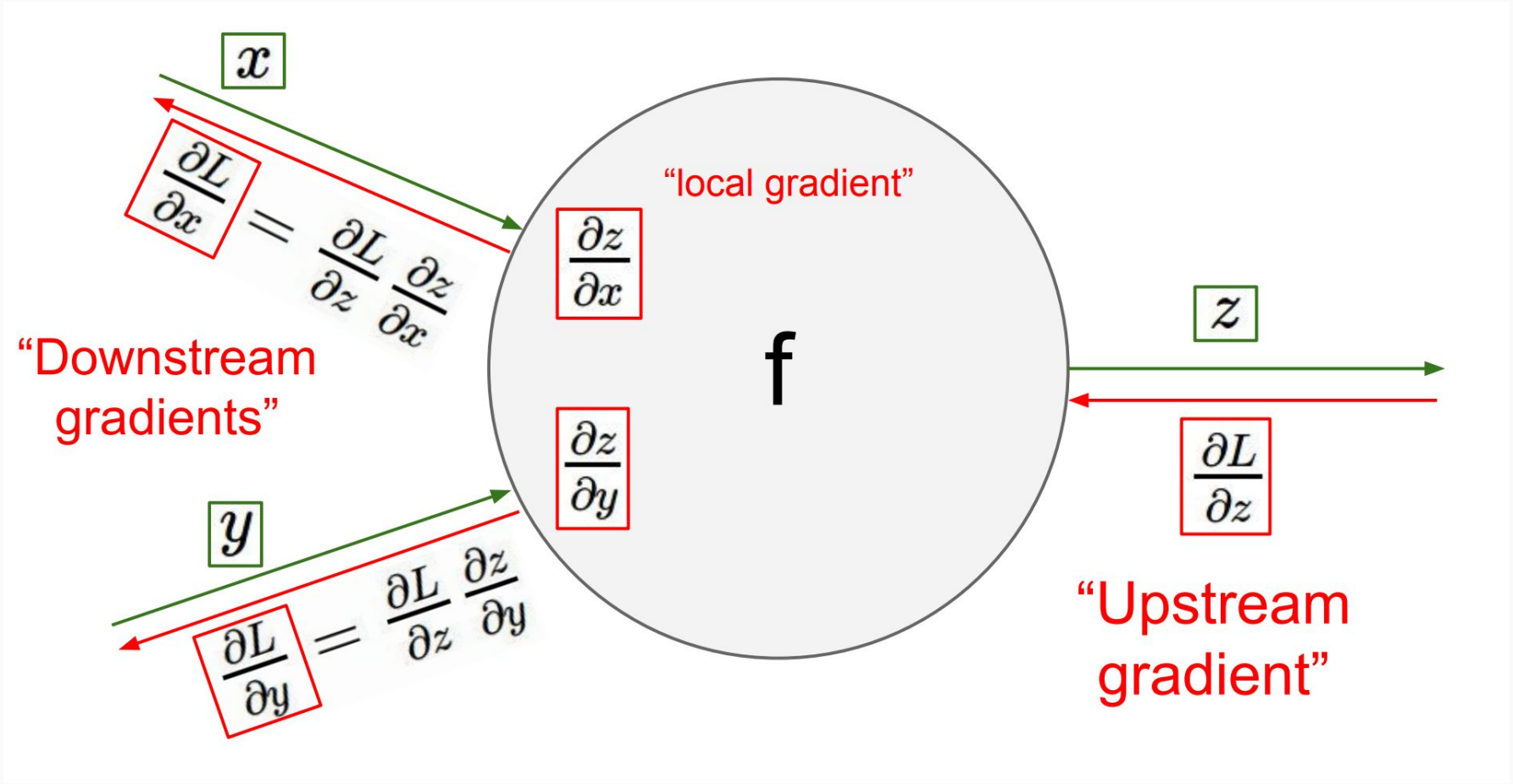
Mean Squared Error (L2 Loss Function)

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

Okay! But how to reduce this loss?

WARNING: Some Calculus Coming


Gradient Descent
Follow the gradient
?
I'm oscillating... what do I do?
Learned Optimizer
?
Aha! I've seen this before...

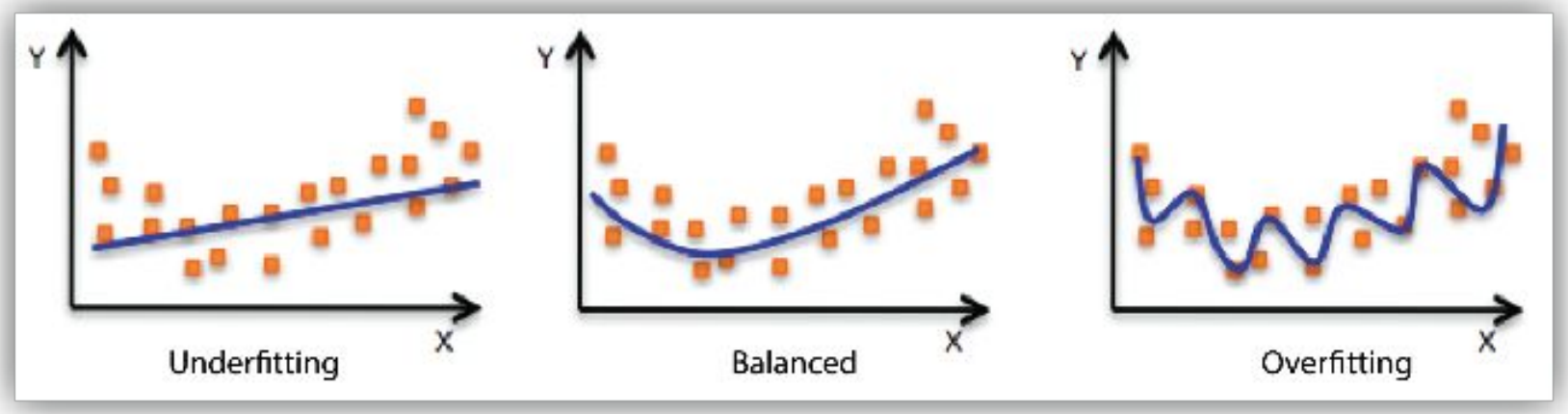

x
$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial x}$
"local gradient"
$\frac{\partial z}{\partial x}$
f
z
"Downstream gradients"
$\frac{\partial z}{\partial y}$
$\frac{\partial L}{\partial z}$
y
$\frac{\partial L}{\partial y} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial y}$
"Upstream gradient"

Linear Regression: Putting it All Together!

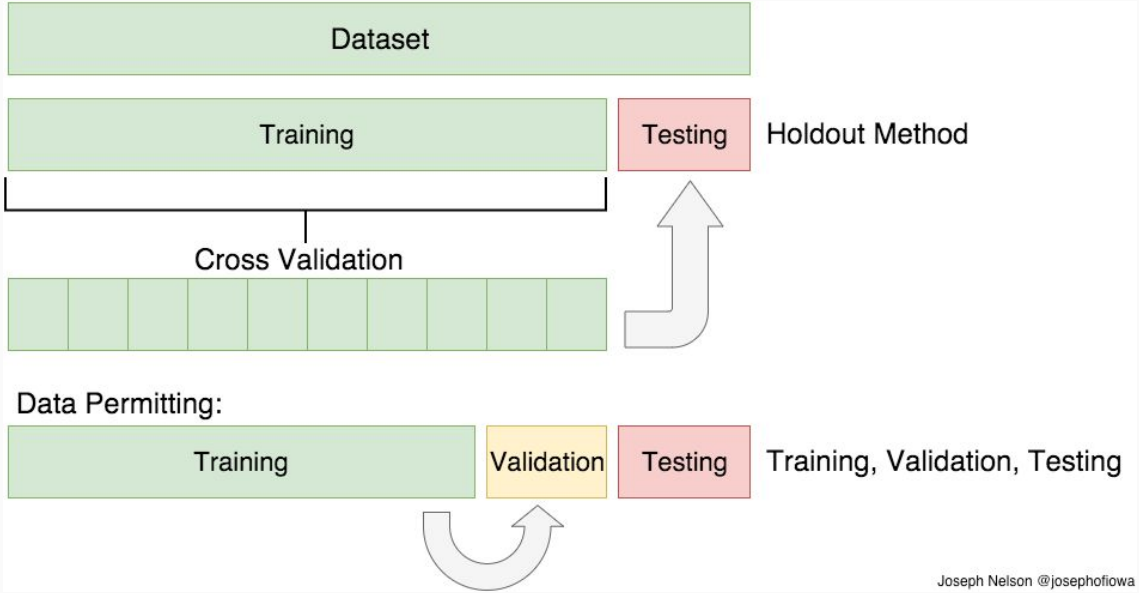
- Start with Randomly Initialized Variables.
- Calculate Loss Function.
- Calculate the gradient and Update those Variables
- And keep on repeating!

Till?

Overfitting v/s Underfitting



Splitting into Training and Testing Data



Demo Time

Reference Material

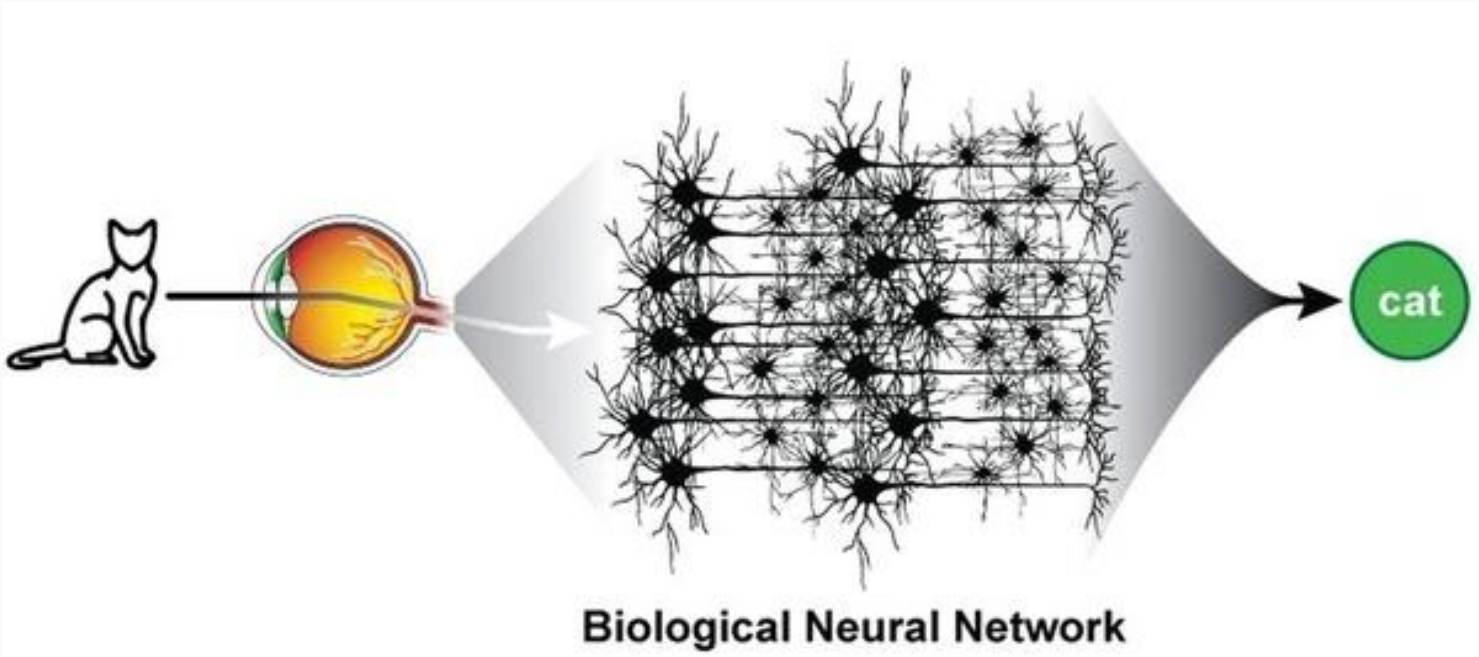
https://community.scaler.com/t/reference-material-for-introduction-to-ml-and-ai-workshop/

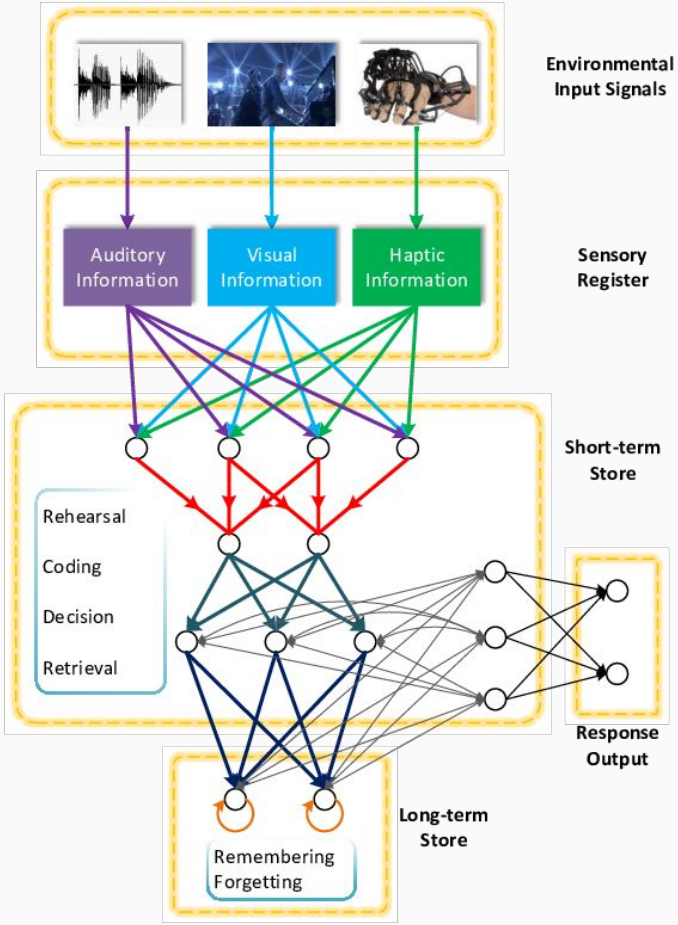
Spread the Knowledge!

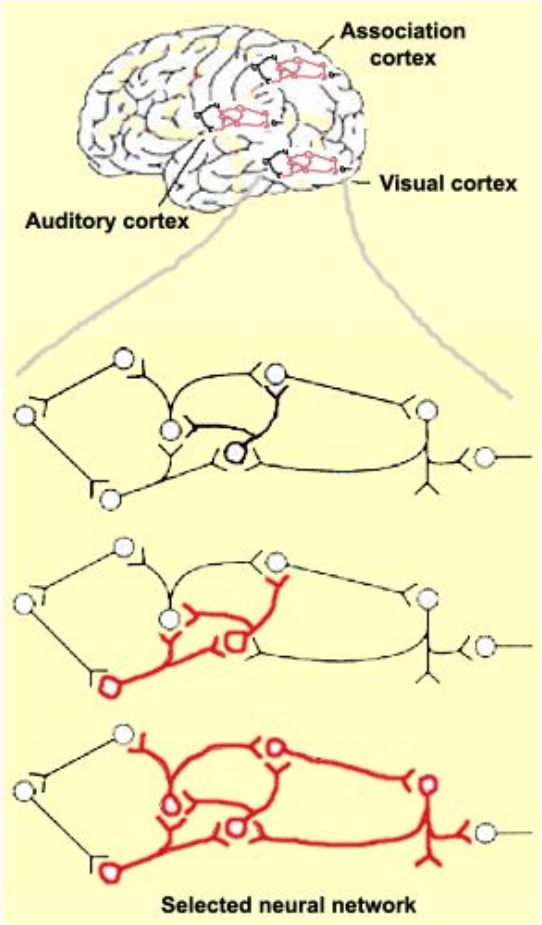
- Write a 200+ words article on any one of the topics discussed over the last 2 days of the Workshop. (List of topics is in next slide)
- Post the article on https://community.scaler.com/c/machine-learning/10 .
- Share your articles with your friends and families, and help them learn ML, the way you just did!
- The top 2 articles for each topic will get cool Scaler Swags!

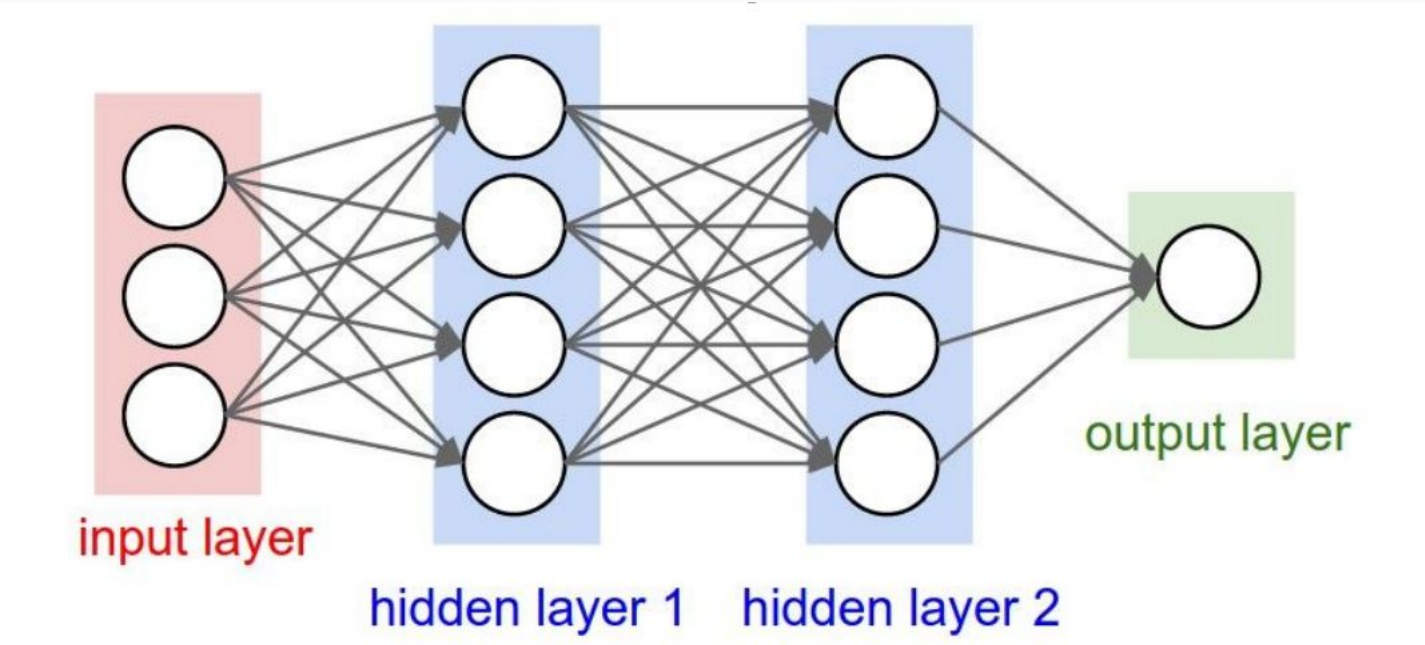
Neural Networks

How human brain works.




Environmental Input Signals
Auditory Information
Visual Information
Haptic Information
Sensory Register
Short-term Store
Rehearsal
Coding
Decision
Retrieval
Response Output
Long-term Store
Remembering
Forgetting


Association cortex
Visual cortex
Auditory cortex
Selected neural network



Why so many Circles?

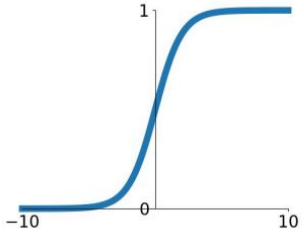
- More random variables!
- More data points => More things being taken into consideration.
- Some similarity with Human brains.
- Neurons that have some specific function.

Layers

- Fine-tune the input weightings until the neural network’s margin of error is minimal.
- Consider each layer to be like a level of abstraction above the previous layer.

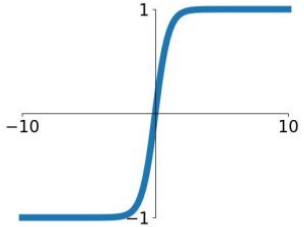
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



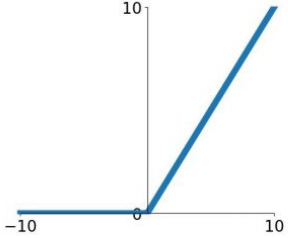
tanh

$$\tanh(x)$$



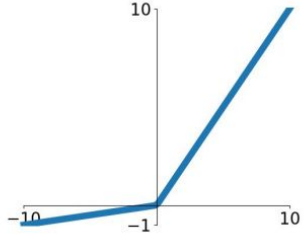
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

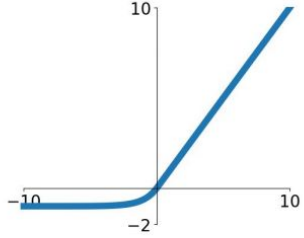


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Intro to Keras Layers

https://keras.io/api/layers/

SCALER EDGE

Dense Layer

Dense layer

`Dense` class

```
tf.keras.layers.Dense(
    units,
    activation=None,
    use_bias=True,
    kernel_initializer="glorot_uniform",
    bias_initializer="zeros",
    kernel_regularizer=None,
    bias_regularizer=None,
    activity_regularizer=None,
    kernel_constraint=None,
    bias_constraint=None,
    **kwargs
)
```

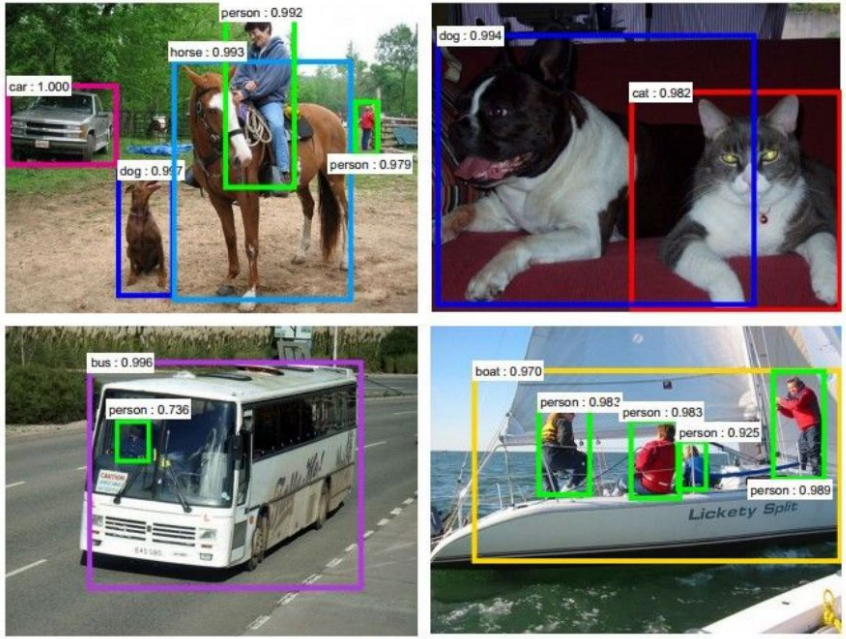
Just your regular densely-connected NN layer.

Keras Activation Functions

- https://keras.io/api/layers/activations
- Let’s have some small demo on it!

Convolutional Neural Networks

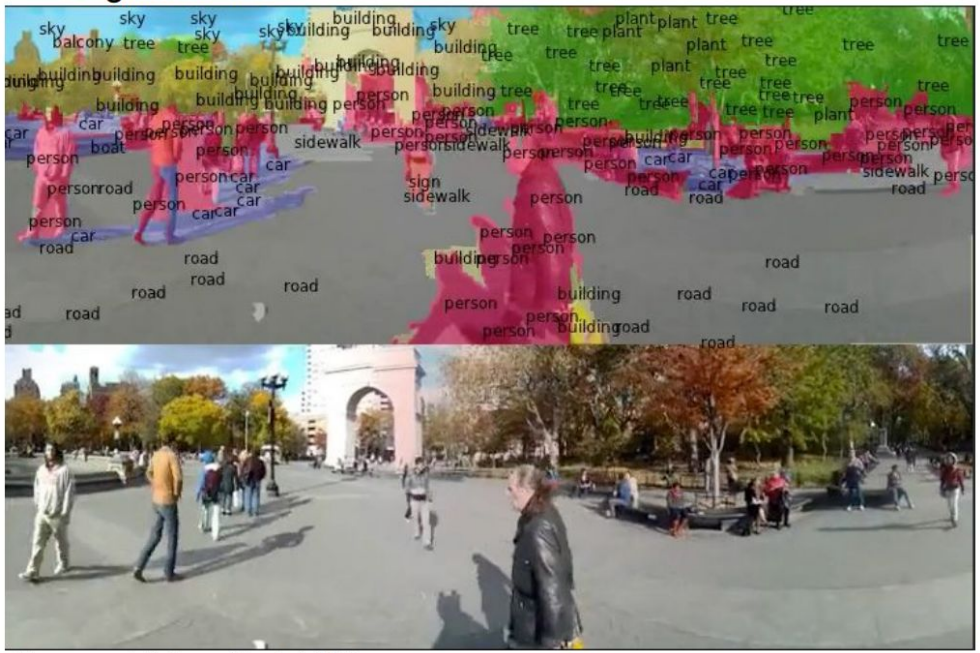
Detection



Figures copyright Shaoqing Ren, Kaiming He, Ross Girschick, Jian Sun, 2015. Reproduced with permission.

[Faster R-CNN: Ren, He, Girshick, Sun 2015]

Segmentation



Figures copyright Clement Farabet, 2012.
Reproduced with permission.

[Farabet et al., 2012]

SCALER EDGE

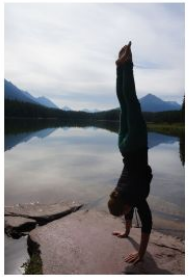
A white teddy bear sitting in the grass

A man in a baseball uniform throwing a ball

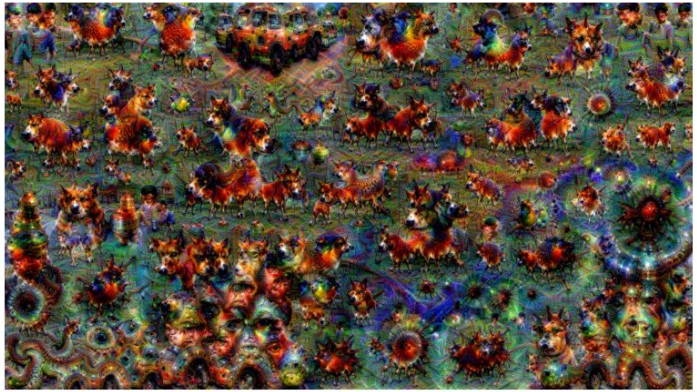
A woman is holding a cat in her hand

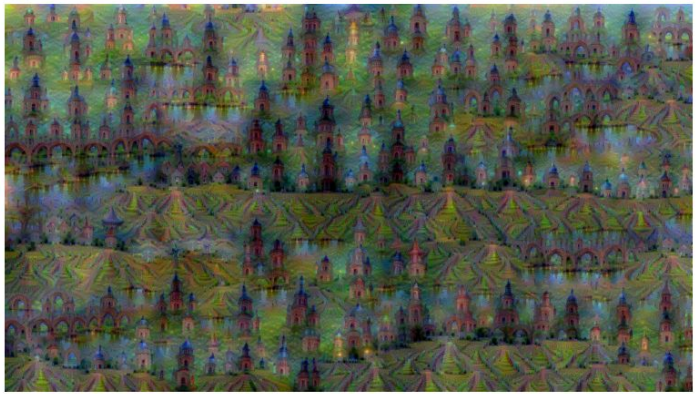
A man riding a wave on top of a surfboard

A cat sitting on a suitcase on the floor



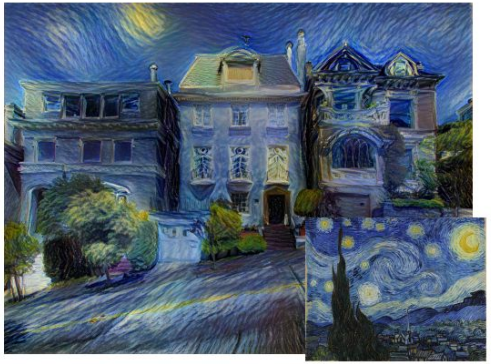
A woman standing on a beach holding a surfboard

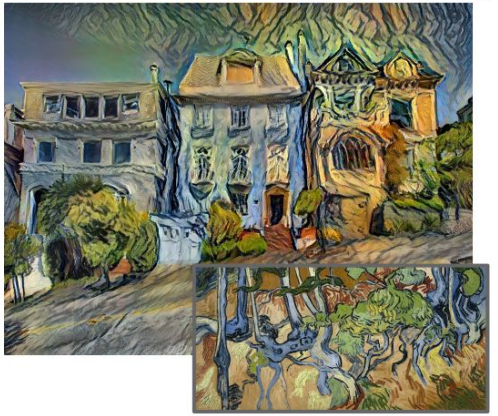




Figures copyright Justin Johnson, 2015. Reproduced with permission. Generated using the Inceptionism approach from a blog post by Google Research.

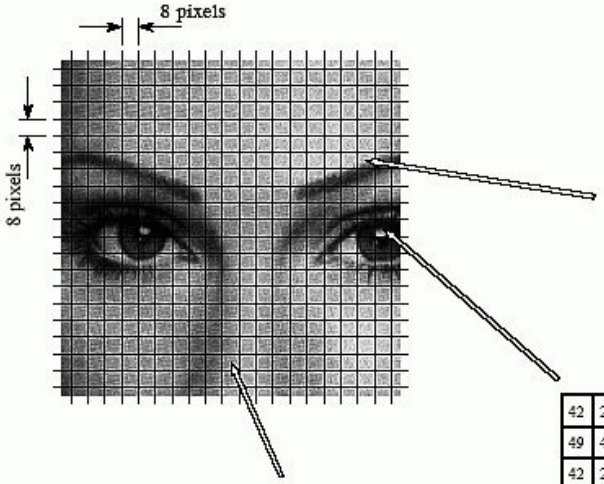
Original image is CC0 public domain
Starry Night and Tree Roots by Van Gogh are in the public domain
Bokeh image is in the public domain
Stylized images copyright Justin Johnson, 2017;
reproduced with permission





Gatys et al, "Image Style Transfer using Convolutional Neural Networks", CVPR 2016
Gatys et al, "Controlling Perceptual Factors in Neural Style Transfer", CVPR 2017

What’s An Image?



231	224	224	217	217	203	189	196
210	217	203	189	203	224	217	224
196	217	210	224	203	203	196	189
210	203	196	203	182	203	182	189
203	224	203	217	196	175	154	140
182	189	168	161	154	126	119	112
175	154	126	105	140	105	119	84
154	98	105	98	105	63	112	84

42	28	35	28	42	49	35	42
49	49	35	28	35	35	35	42
42	21	21	28	42	35	42	28
21	35	35	42	42	28	28	14
56	70	77	84	91	28	28	21
70	126	133	147	161	91	35	14
126	203	189	182	175	175	35	21
49	189	245	210	182	84	21	35

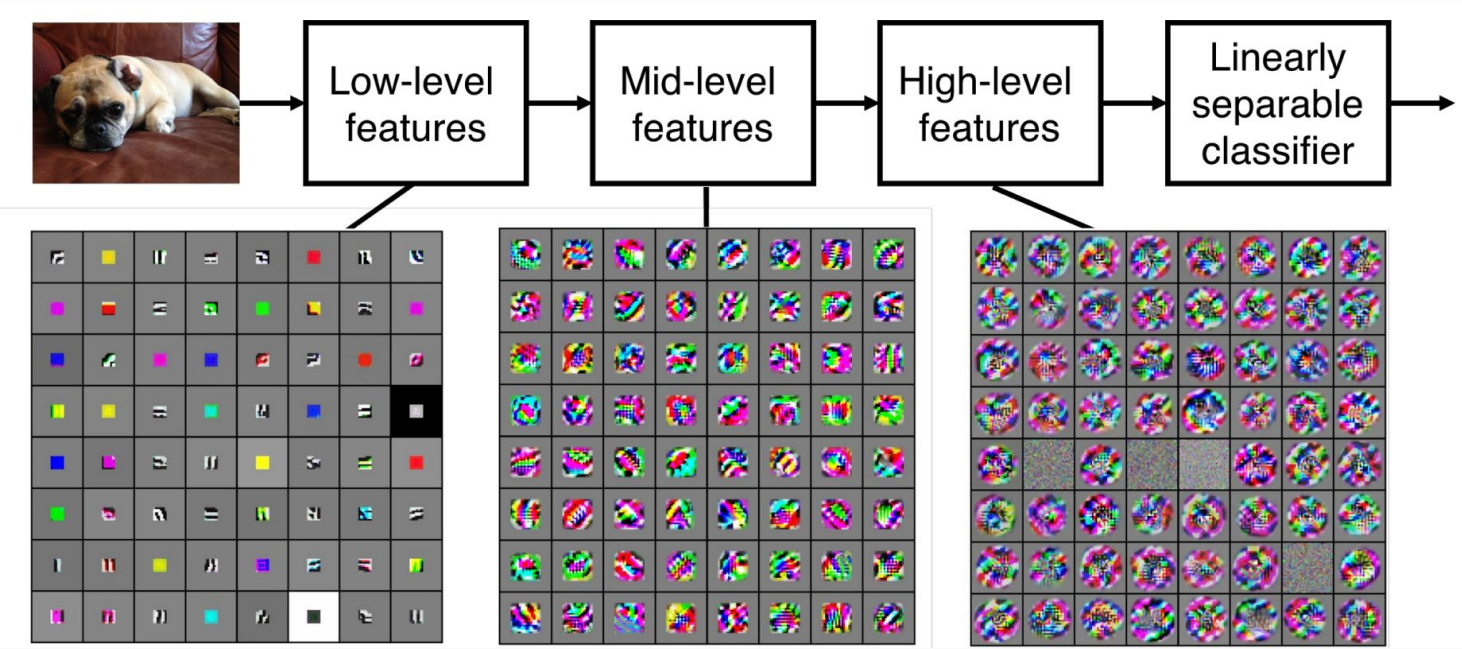
154	154	175	182	189	168	217	175
154	147	168	154	168	168	196	175
175	154	203	175	189	182	196	182
175	168	168	168	140	175	168	203
133	168	154	196	175	189	203	154
168	161	161	168	154	154	189	189
147	161	175	182	189	175	217	175
175	175	203	175	189	175	175	182

FIGURE 27-9
JPEG image division. JPEG transform compression starts by breaking the image into 8×8 groups, each containing 64 pixels. Three of these 8×8 groups are enlarged in this figure, showing the values of the individual pixels, a single byte value between 0 and 255.

How Humans See?

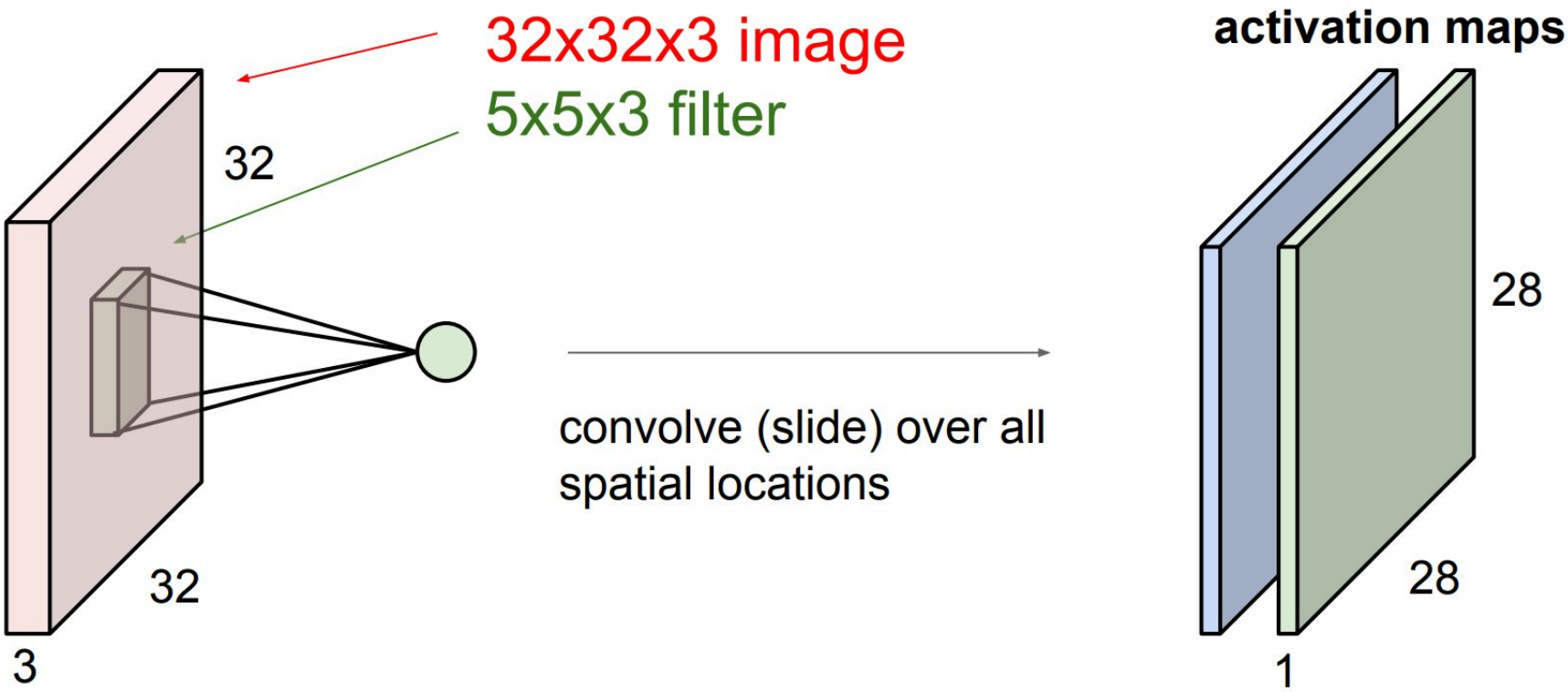
- Close your eyes and try to open eyes slooooowly!
- How do you start seeing?
- First you see some patterns, then more specific patterns, then more, then more and finally you see everything clearly.

And this is how CNNs see!



SCALER EDGE

But HOW?


32x32x3 image
5x5x3 filter
32
32
3
convolve (slide) over all
spatial locations
activation maps
28
28
1

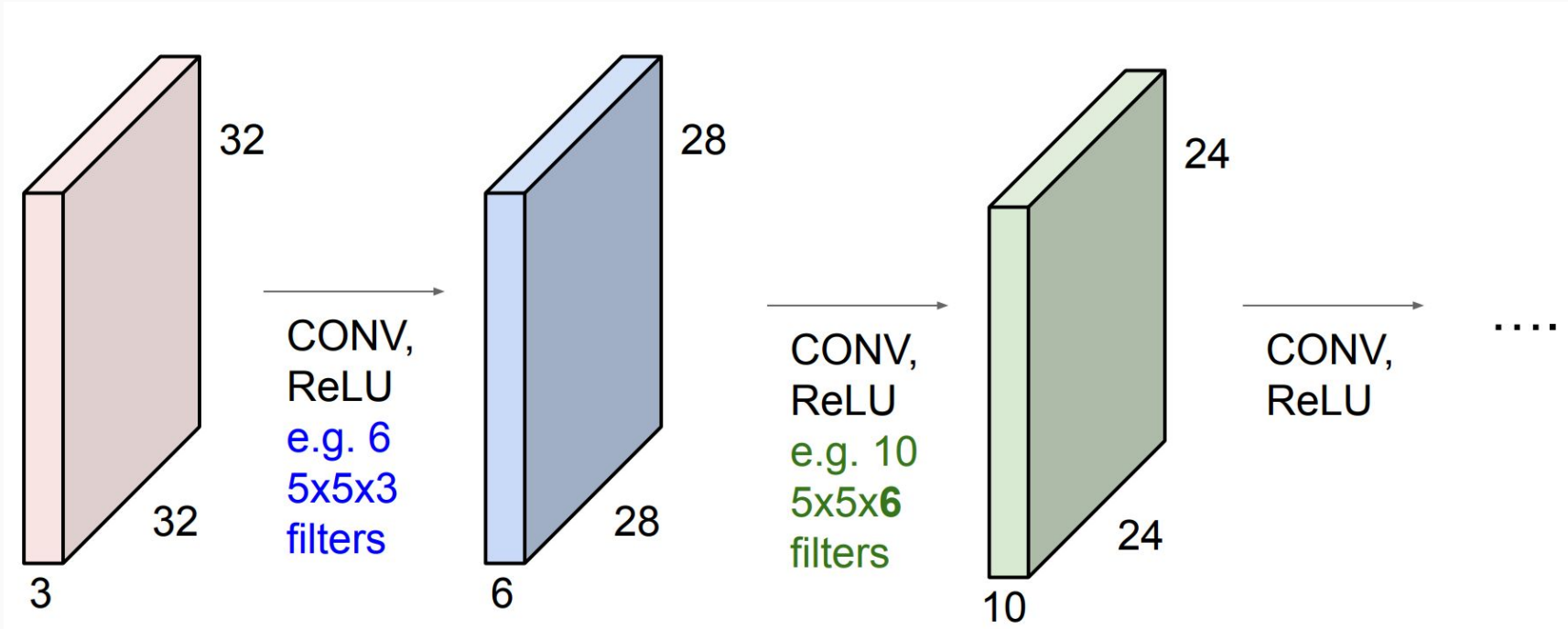
Some Terminology

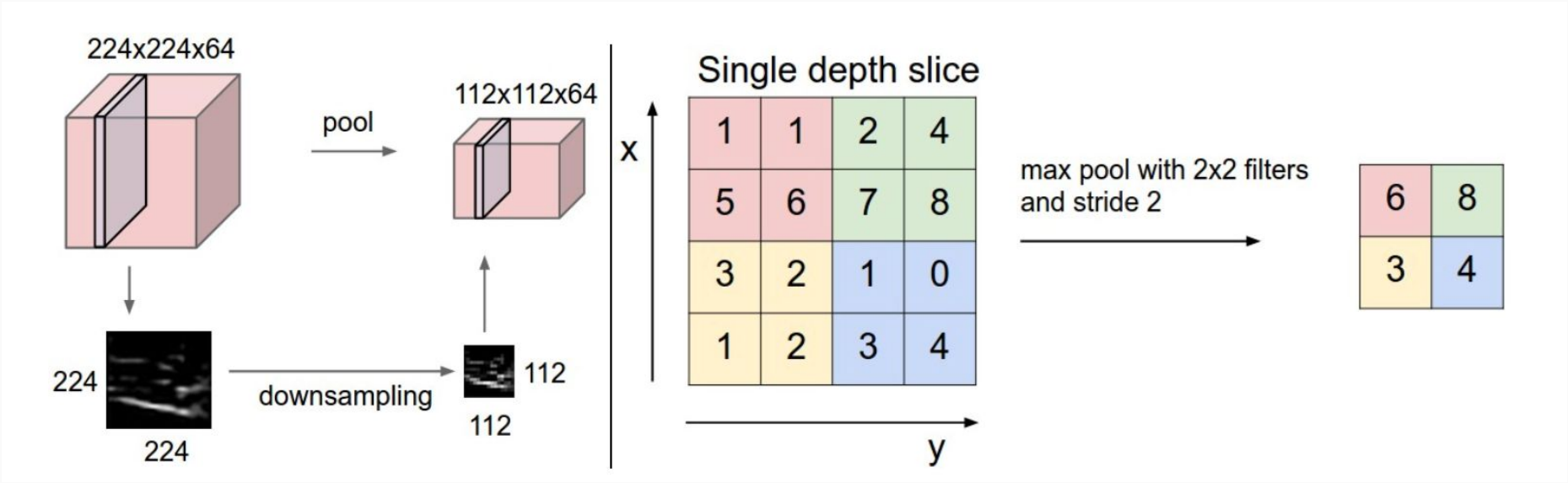
Kernel

- The matrix that is convolving over the image.

Stride

- Amount of shift happening in the the convolving matrix at each step.
- Most often: (1,1)


32
32
3
CONV,
ReLU
e.g. 6
5x5x3
filters
28
28
6
CONV,
ReLU
e.g. 10
5x5x6
filters
24
24
10
CONV,
ReLU
....


224x224x64
pool
112x112x64
224
downsampling
112
112
Single depth slice
x
y
1 1 2 4
5 6 7 8
3 2 1 0
1 2 3 4
max pool with 2x2 filters
and stride 2
6 8
3 4

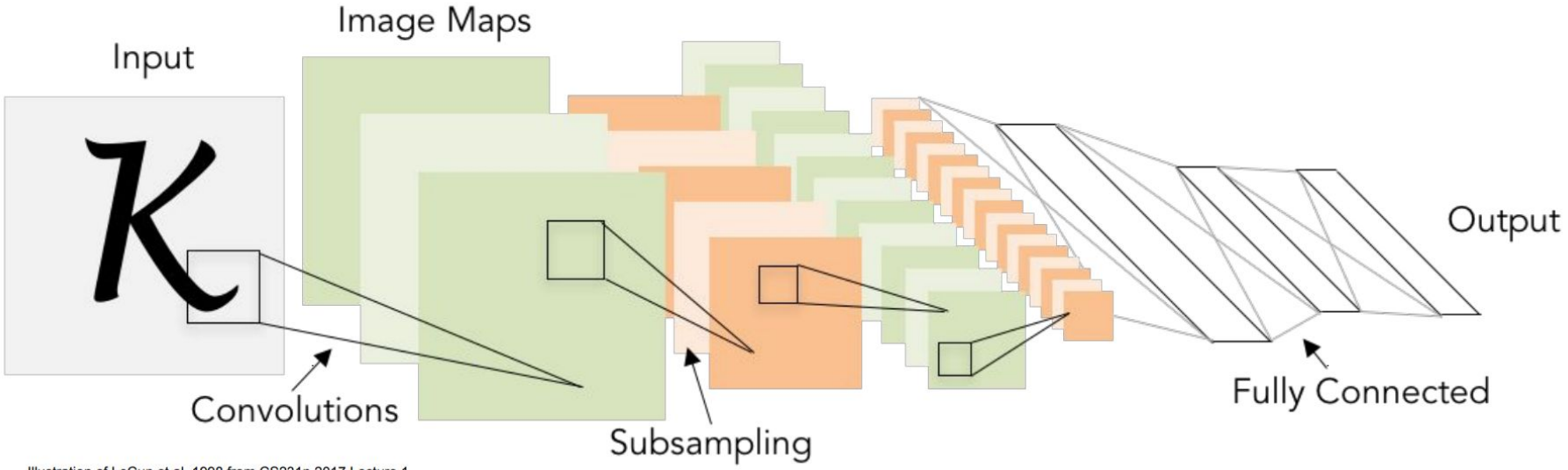


Illustration of LeCun et al. 1998 from CS231n 2017 Lecture 1

How Keras supports CNN Layers

Convolution layers

- Conv1D layer
- Conv2D layer
- Conv3D layer
- SeparableConv1D layer
- SeparableConv2D layer
- DepthwiseConv2D layer
- Conv2DTranspose layer
- Conv3DTranspose layer

Pooling layers

- MaxPooling1D layer
- MaxPooling2D layer
- MaxPooling3D layer
- AveragePooling1D layer
- AveragePooling2D layer
- AveragePooling3D layer
- GlobalMaxPooling1D layer
- GlobalMaxPooling2D layer
- GlobalMaxPooling3D layer
- GlobalAveragePooling1D layer
- GlobalAveragePooling2D layer
- GlobalAveragePooling3D layer

layers.Conv2D

Conv2D layer

Conv2D class

```
tf.keras.layers.Conv2D(
    filters,
    kernel_size,
    strides=(1, 1),
    padding="valid",
    data_format=None,
    dilation_rate=(1, 1),
    groups=1,
    activation=None,
    use_bias=True,
    kernel_initializer="glorot_uniform",
    bias_initializer="zeros",
    kernel_regularizer=None,
    bias_regularizer=None,
    activity_regularizer=None,
    kernel_constraint=None,
    bias_constraint=None,
    **kwargs
)
```

2D convolution layer (e.g. spatial convolution over images).

layers.MaxPooling2D

MaxPooling2D layer

`MaxPooling2D` class

```
tf.keras.layers.MaxPooling2D(
    pool_size=(2, 2), strides=None, padding="valid", data_format=None, **kwargs
)
```

Max pooling operation for 2D spatial data.

Downsamples the input representation by taking the maximum value over the window defined by `pool_size` for each dimension along the features axis. The window is shifted by `strides` in each dimension. The resulting output when using "valid" padding option has a shape(number of rows or columns) of: `output_shape = (input_shape - pool_size + 1) / strides)`

The resulting output shape when using the "same" padding option is: `output_shape = input_shape / strides`

layers.Flatten

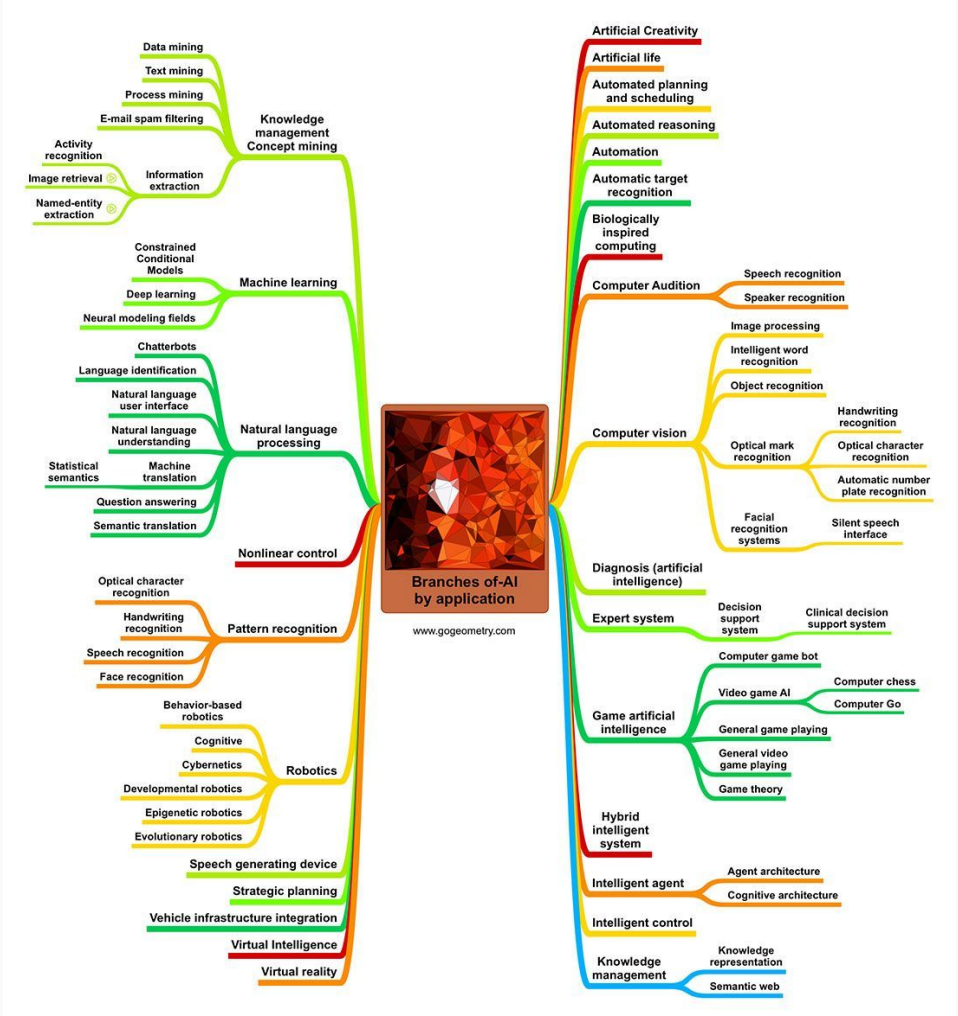
```
>>> model = tf.keras.Sequential()
>>> model.add(tf.keras.layers.Conv2D(64, 3, 3, input_shape=(3, 32, 32)))
>>> model.output_shape
(None, 1, 10, 64)
```

```
>>> model.add(Flatten())
>>> model.output_shape
(None, 640)
```

Demo Time

Spread the Knowledge!

- Write a 200+ words article on any one of the topics discussed over the last 2 days of the Workshop. (List of topics is in next slide.)
- Post the article on https://community.scaler.com/c/machine-learning/10 .
- Share your articles with your friends and families, and help them learn ML, the way you just did!
- The top 2 articles for each topic will get cool Scaler Swags!


Branches of-AI by application
www.gogeometry.com
Knowledge management
Concept mining
Data mining
Text mining
Process mining
E-mail spam filtering
Information extraction
Activity recognition
Image retrieval
Named-entity extraction
Machine learning
Constrained Conditional Models
Deep learning
Neural modeling fields
Natural language processing
Chatterbots
Language identification
Natural language user interface
Natural language understanding
Machine translation
Statistical semantics
Question answering
Semantic translation
Nonlinear control
Pattern recognition
Optical character recognition
Handwriting recognition
Speech recognition
Face recognition
Robotics
Behavior-based robotics
Cognitive
Cybernetics
Developmental robotics
Epigenetic robotics
Evolutionary robotics
Speech generating device
Strategic planning
Vehicle infrastructure integration
Virtual Intelligence
Virtual reality
Artificial Creativity
Artificial life
Automated planning and scheduling
Automated reasoning
Automation
Automatic target recognition
Biologically inspired computing
Computer Audition
Speech recognition
Speaker recognition
Computer vision
Image processing
Intelligent word recognition
Object recognition
Optical mark recognition
Handwriting recognition
Optical character recognition
Automatic number plate recognition
Facial recognition systems
Silent speech interface
Diagnosis (artificial intelligence)
Expert system
Decision support system
Clinical decision support system
Game artificial intelligence
Computer game bot
Video game AI
Computer chess
Computer Go
General game playing
General video game playing
Game theory
Hybrid intelligent system
Intelligent agent
Agent architecture
Cognitive architecture
Intelligent control
Knowledge management
Knowledge representation
Semantic web

SCALER EDGE