Introduction to Machine Learning

Convolutional Neural Networks,

Image Processing,

and

TensorFlow

Deep Learning

- Image processing
  - Face recognition
  - Handwriting recognition
  - Video segmentation

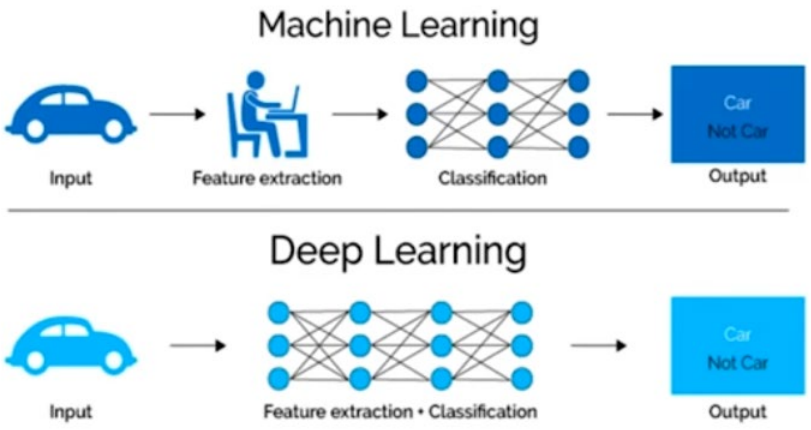
Deep Learning

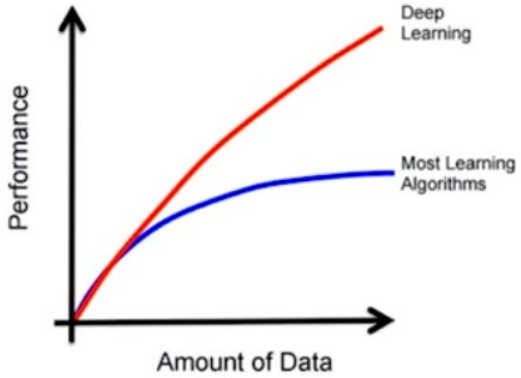
- Image processing
  - Face recognition
  - Handwriting recognition
  - Video segmentation

Deep Learning

- Image processing
  - Face recognition
  - Handwriting recognition
  - Video segmentation
- Speech recognition
- Machine translation
- Game playing
- Auto driving

Why deep learning?





Why deep learning?

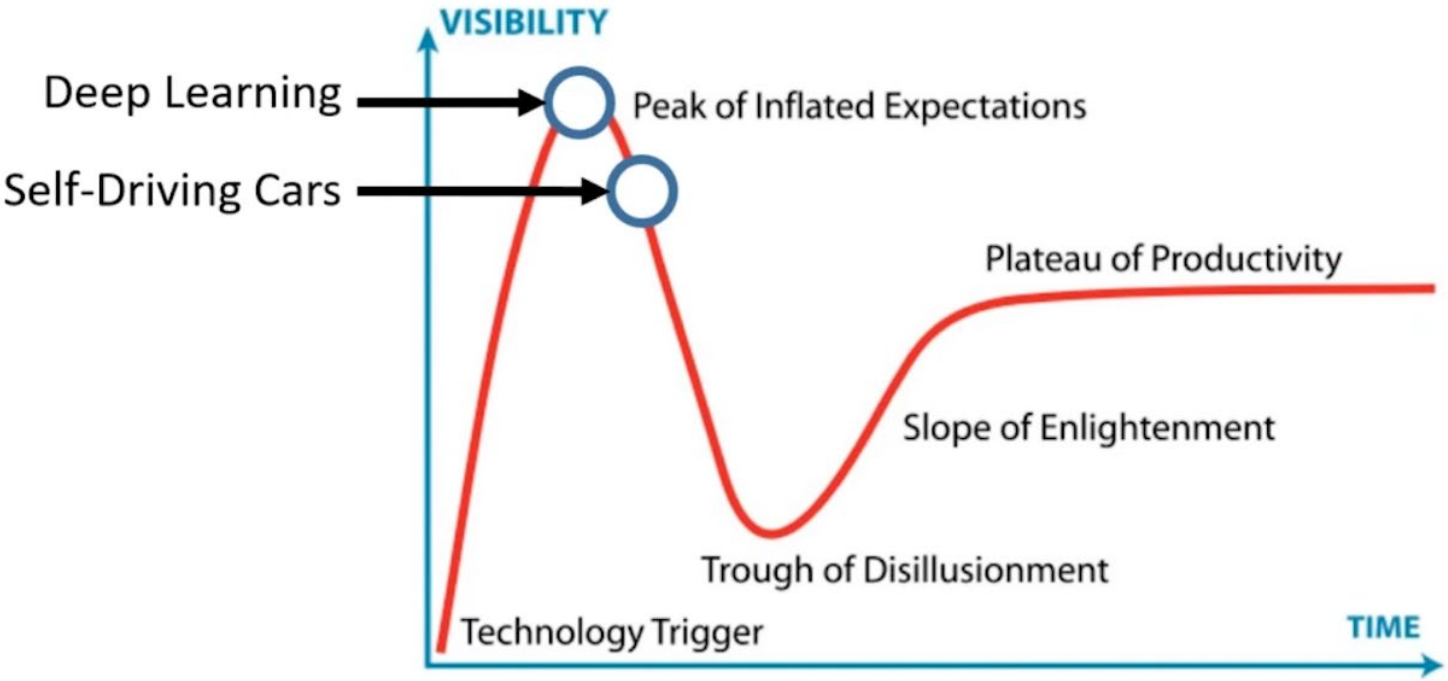


Image analysis

Oval-shaped white blob (body)

Round, elongated oval with orange protuberance

Long white rectangular shape (neck)

Image analysis

Round, elongated head with orange or black beak

Oval-shaped white body with or without large white symmetric blobs (wings)

Long white neck, square shape

Image analysis

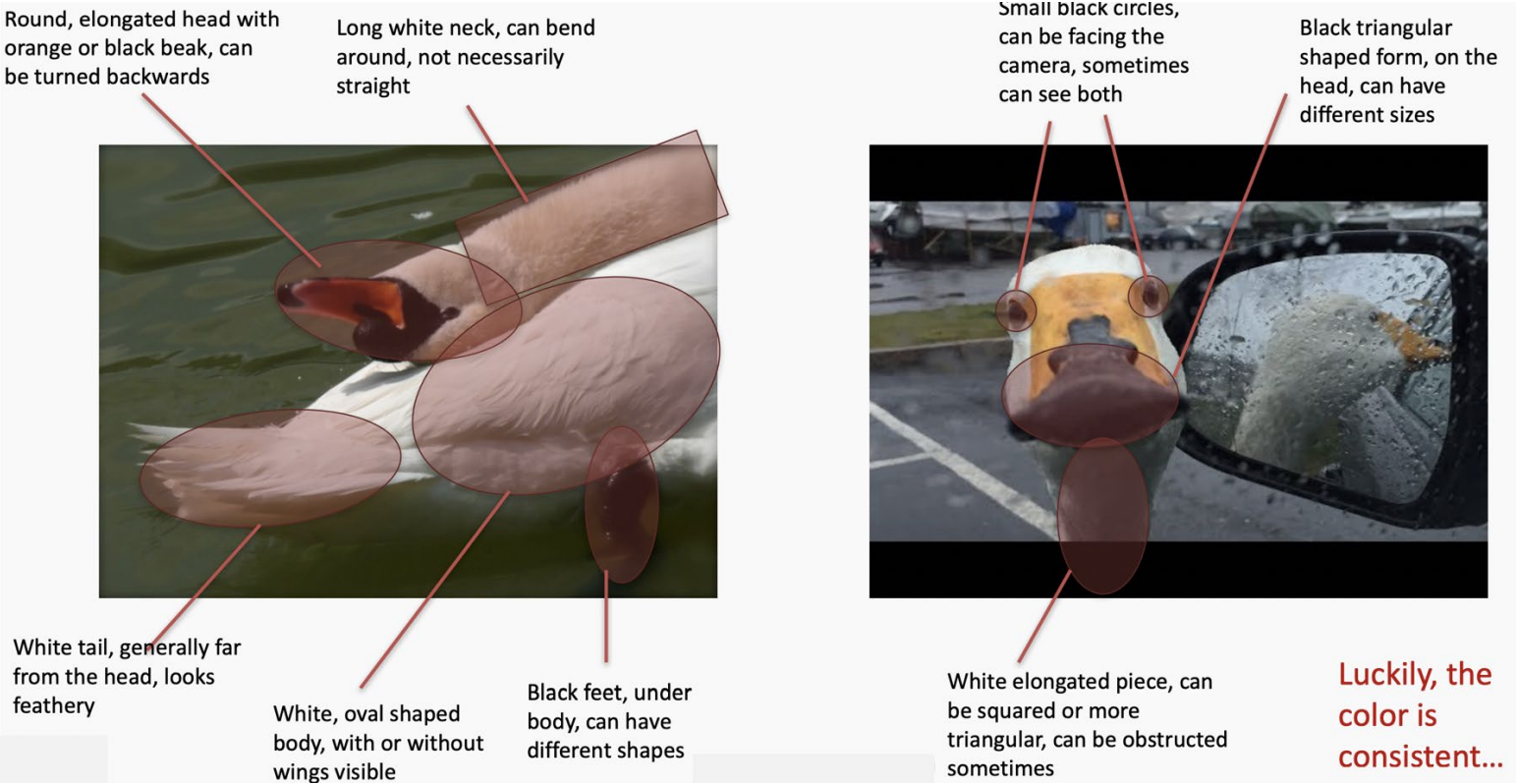
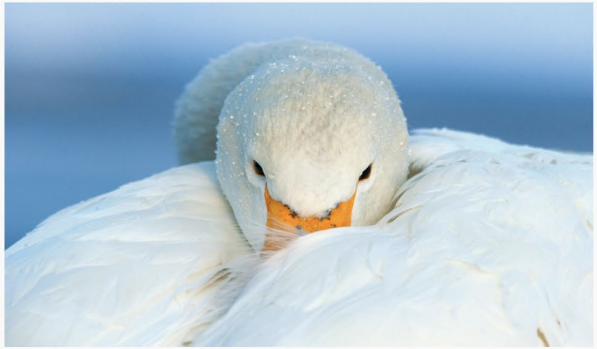


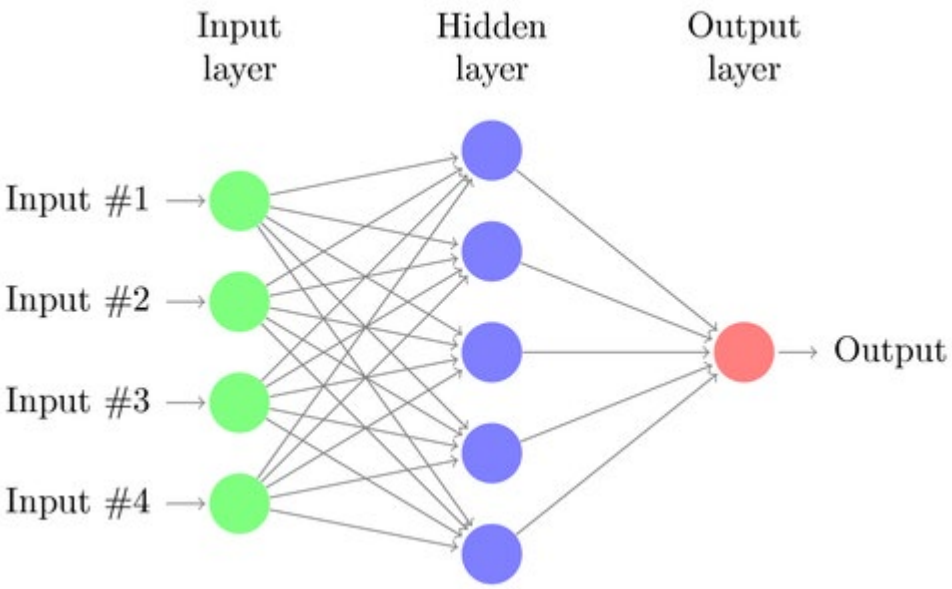
Image analysis

Image analysis

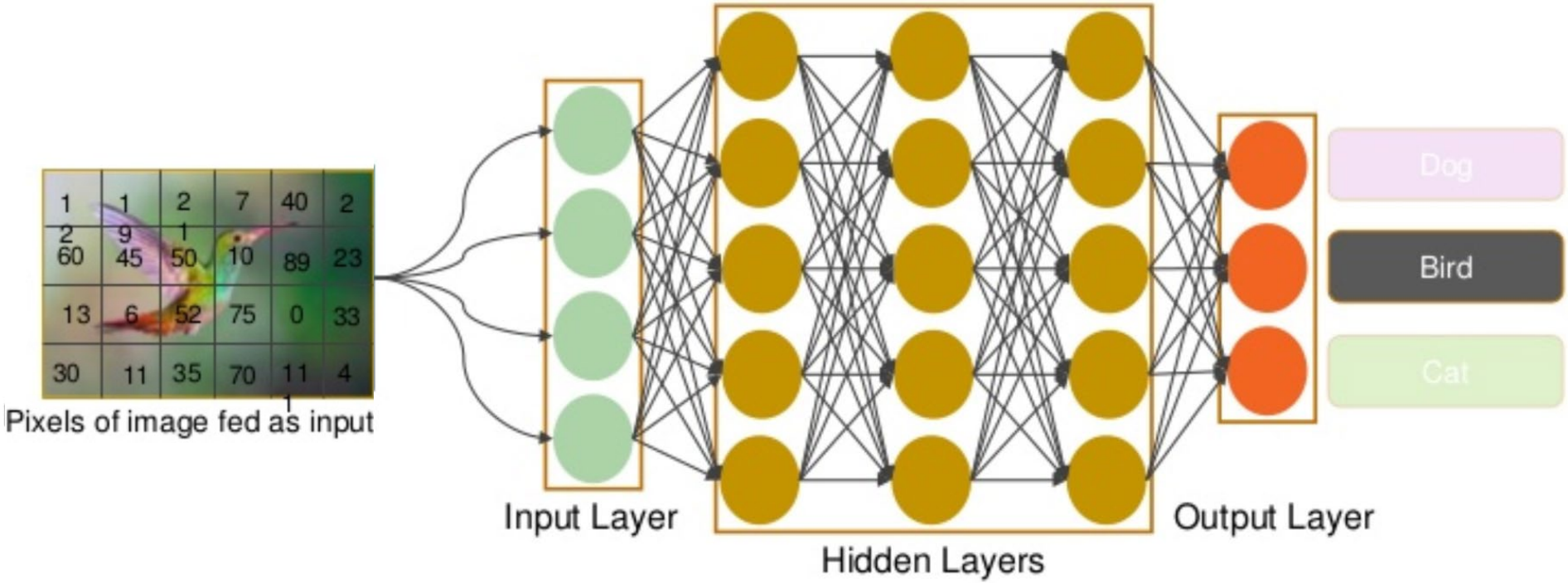


Man in swan tent photographing swans

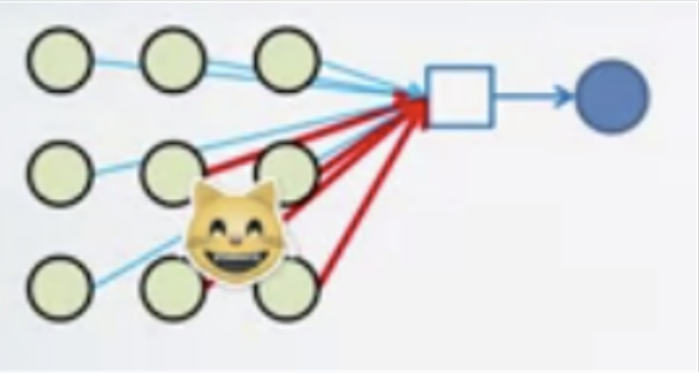
Traditional NNs



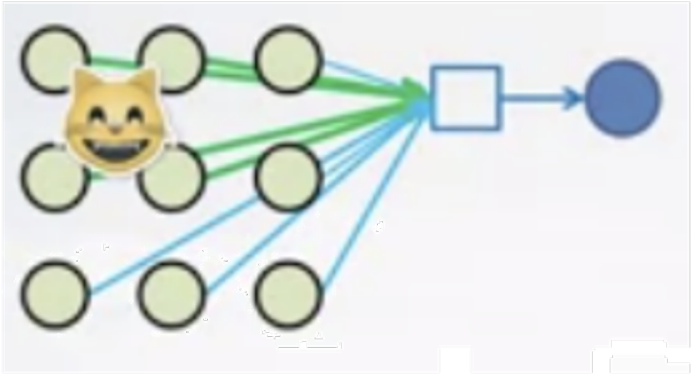
CNNs and image processing



Traditional NNs



In this case, the red weights will be modified to better recognize cats



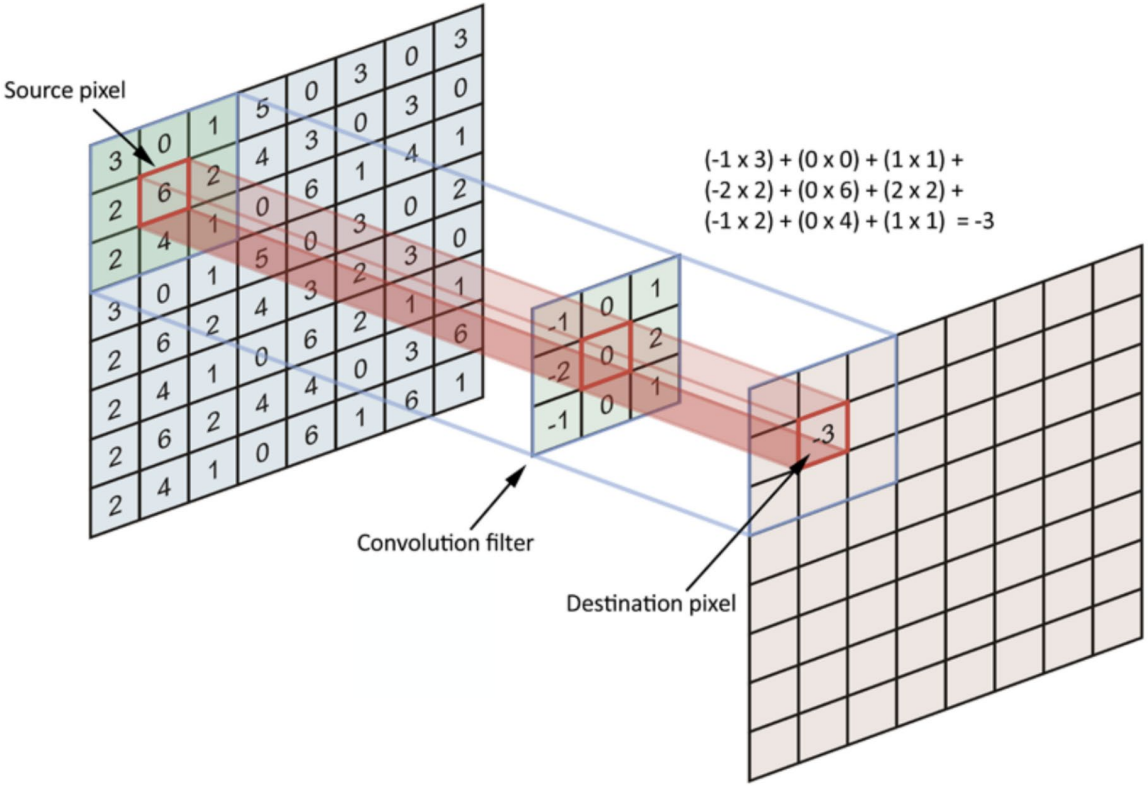
In this case, the green weights will be modified.

Convolutional neural network

Nearby pixels are more strongly related than distant ones.

Objects are built up out of smaller parts.

Convolutional neural network



Convolutional neural network

Edge detection

Kernel

$$* \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} =$$

Sharpen

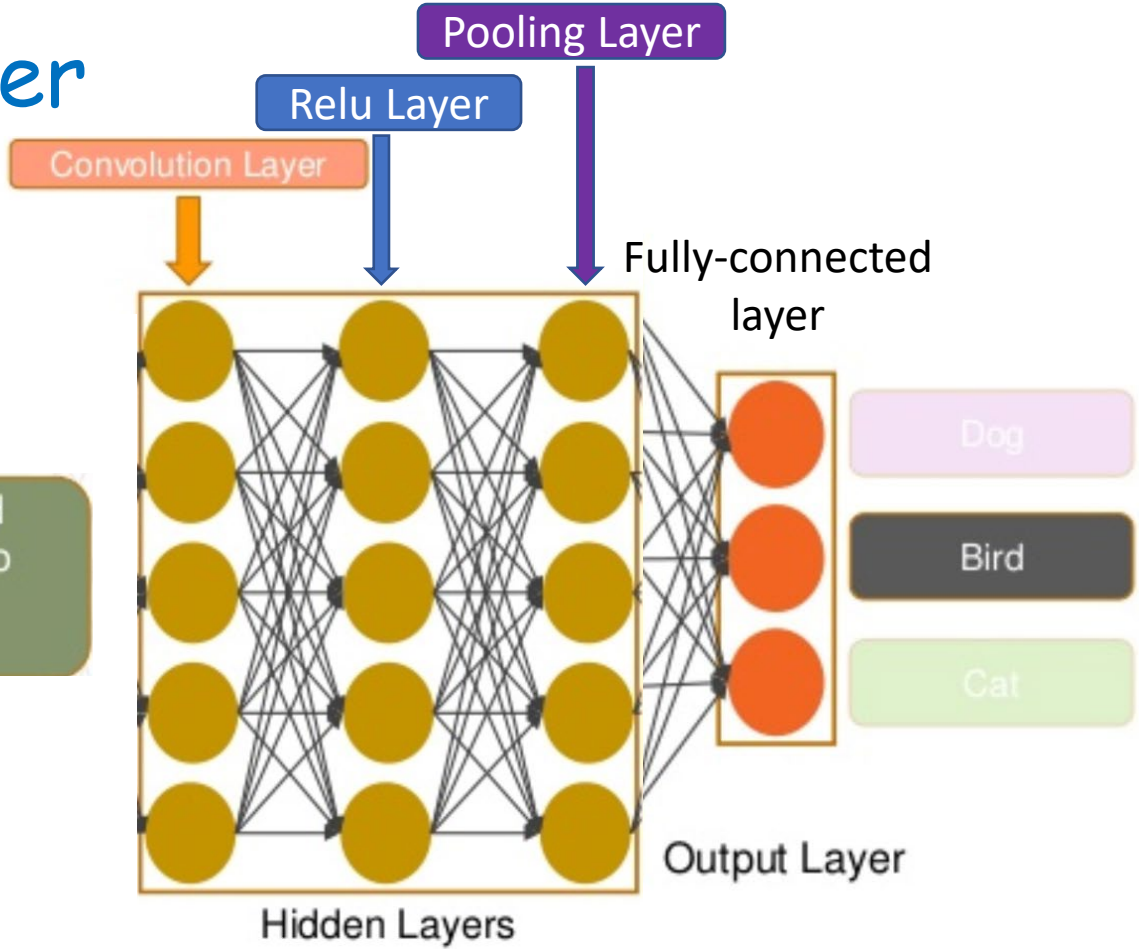
$$* \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} =$$

Convolution layer

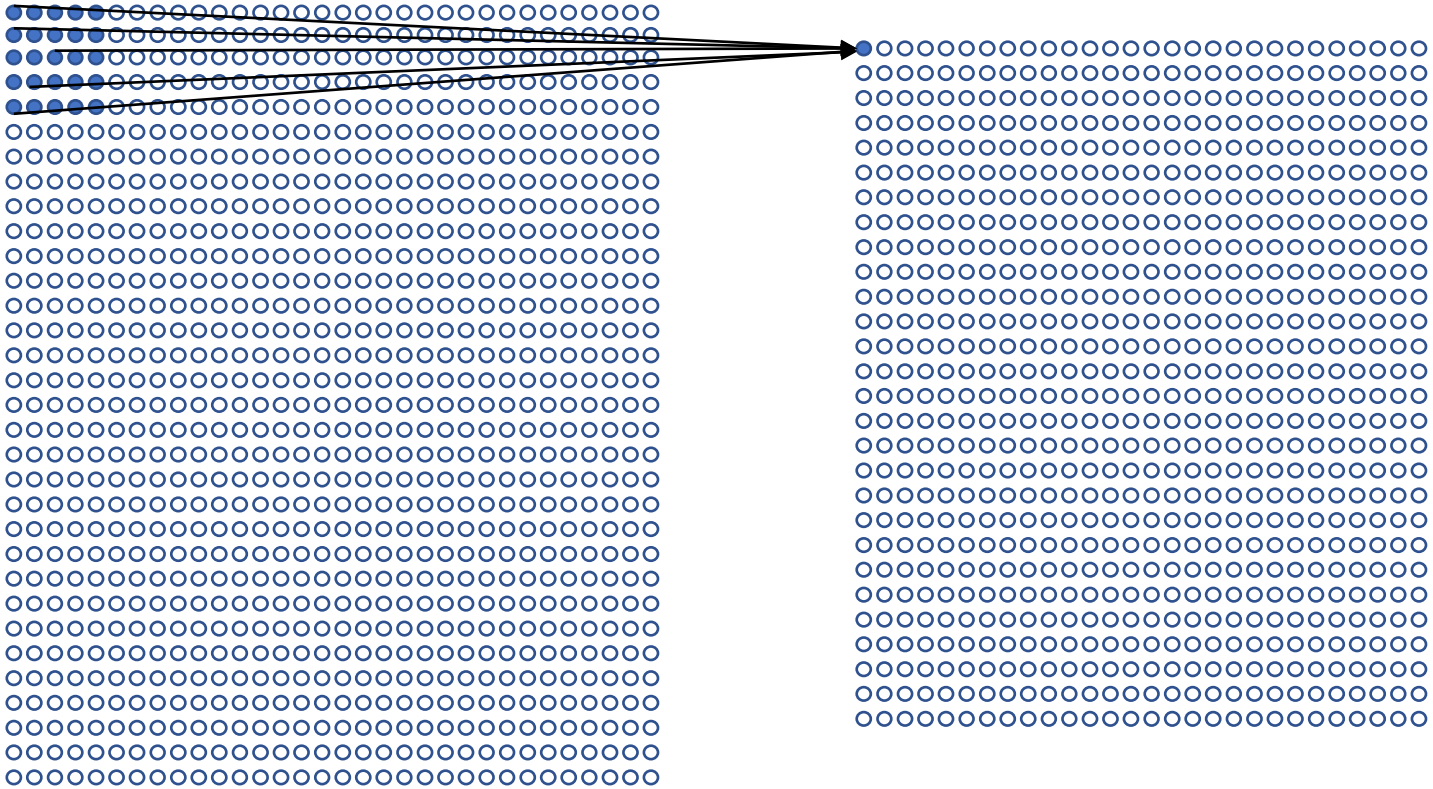
Matrix Filter

1	0	1
0	1	0
1	0	1

This layer uses a matrix filter and performs convolution operation to detect patterns in the image



Convolution layer



Convolution layer

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

1	0	1
0	1	0
1	0	1

$1_{\times 1}$	$1_{\times 0}$	$1_{\times 1}$	0	0
$0_{\times 0}$	$1_{\times 1}$	$1_{\times 0}$	1	0
$0_{\times 1}$	$0_{\times 0}$	$1_{\times 1}$	1	1
0	0	1	1	0
0	1	1	0	0

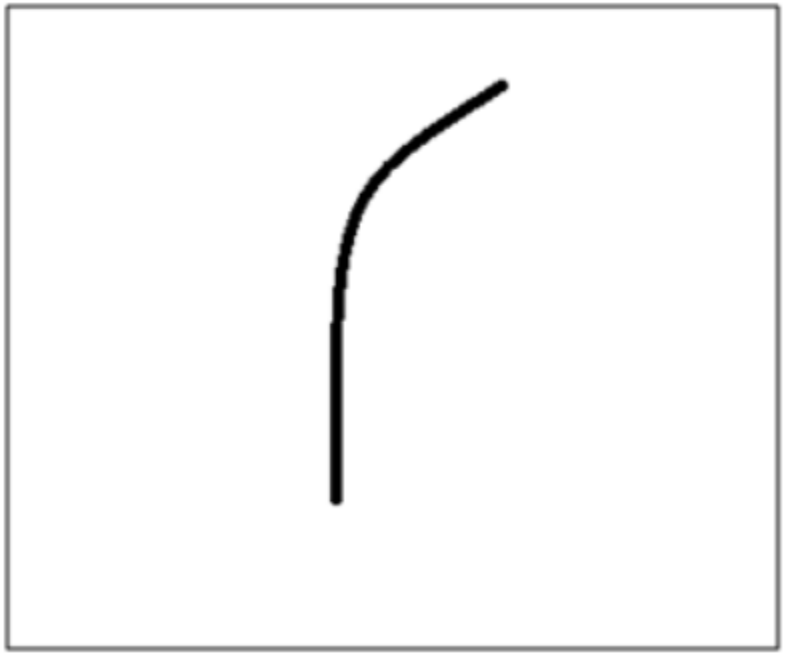
Image

4		

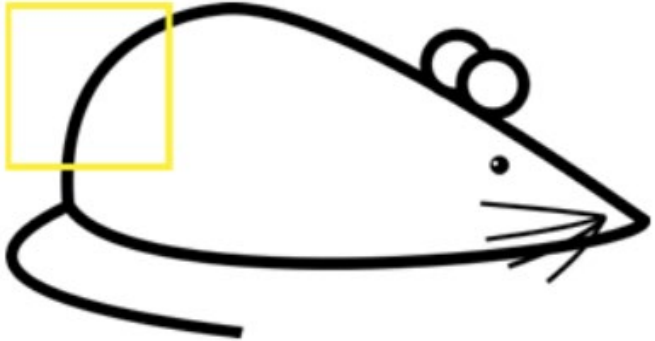
Convolved Feature

What is the convolution layer doing?

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0



What is the convolution layer doing?

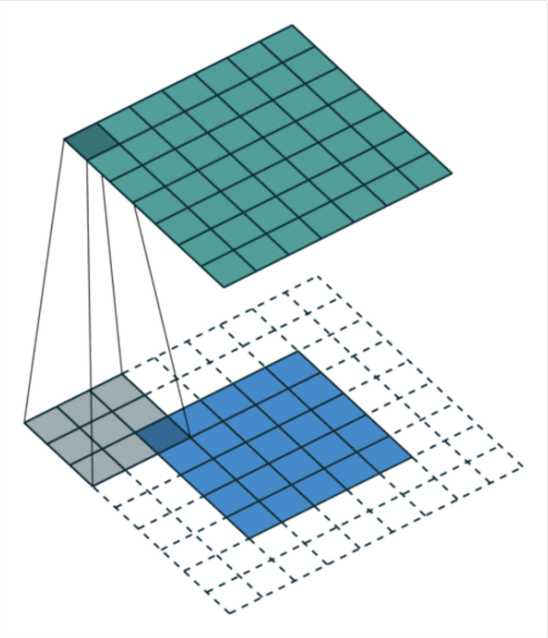


0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

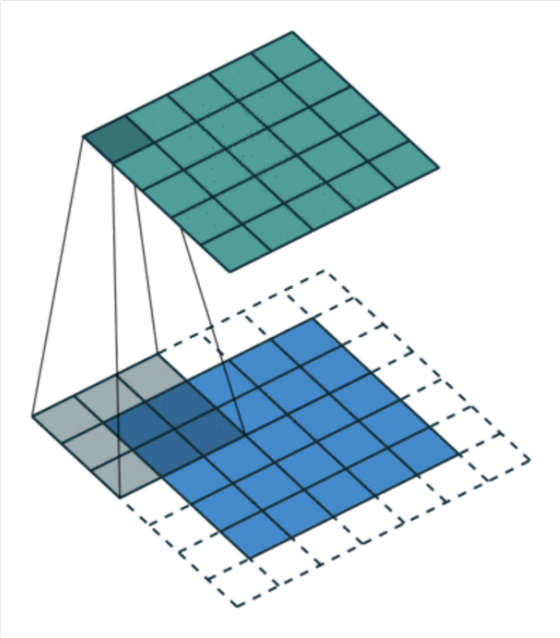
*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Padding



Full padding. Introduces zeros such that all pixels are visited the same amount of times by the filter. Increases size of output.

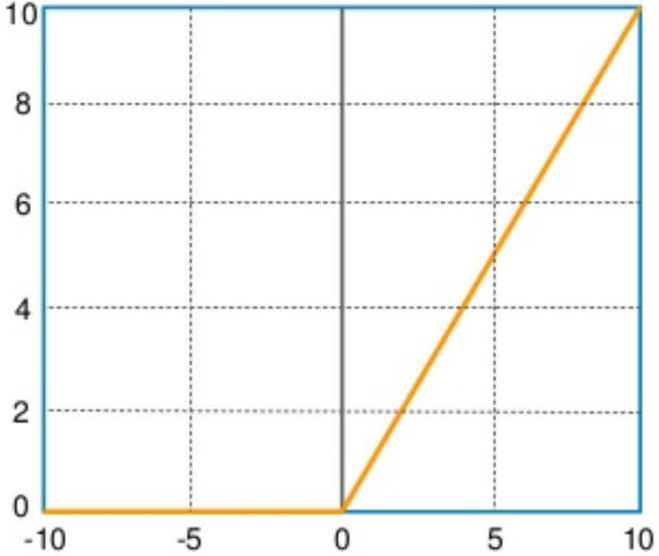


Same padding. Ensures that the output has the same size as the input.

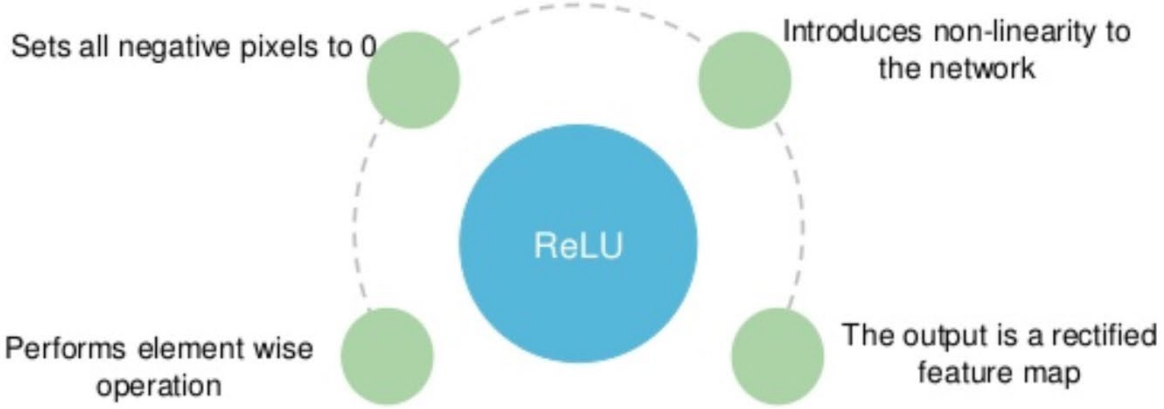
Example

ReLU layer

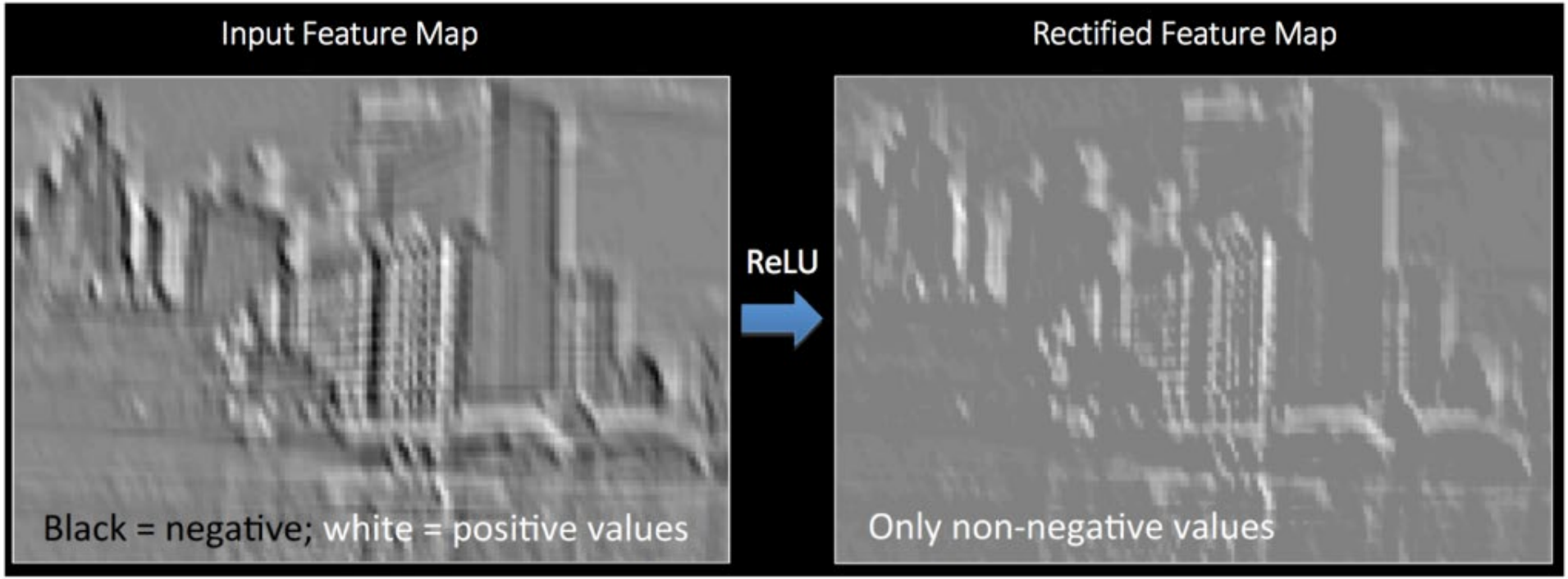
$$R(z) = \max(0, z)$$



$$\sum_{i=1}^{d} w_i \, x_i$$

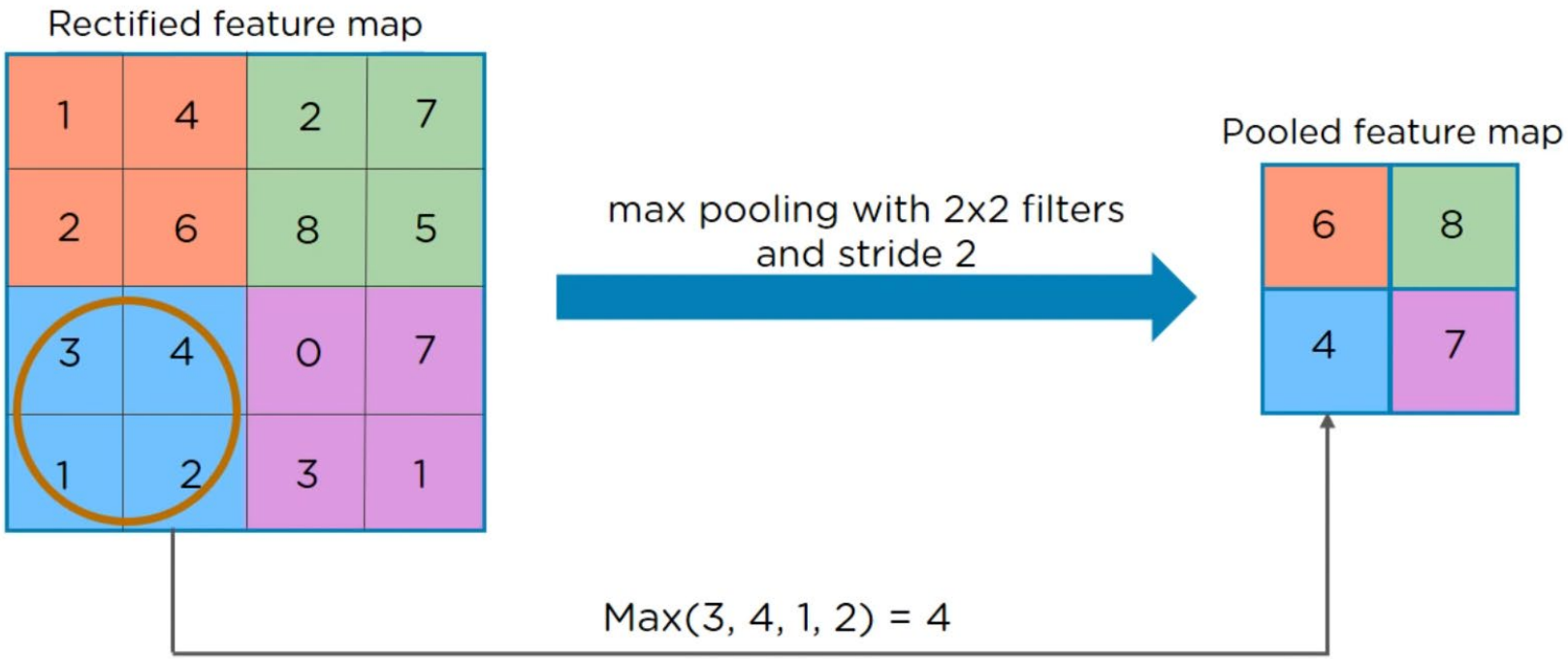


ReLU layer effect

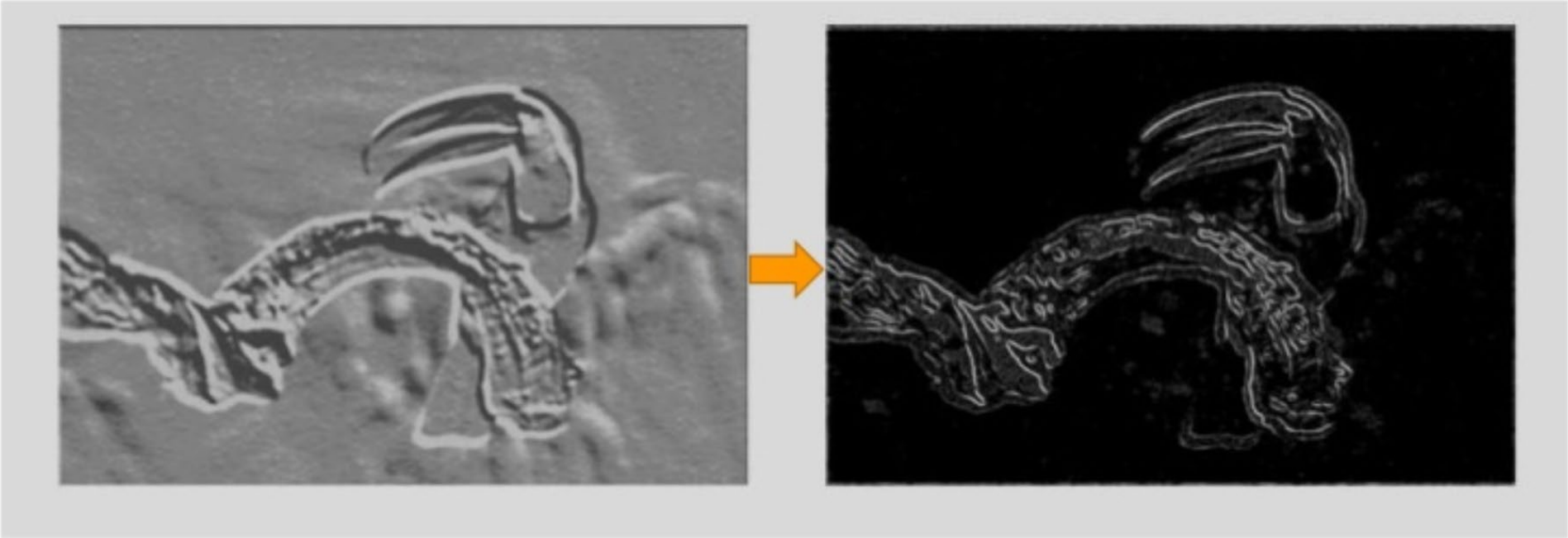


ReLU layer effect

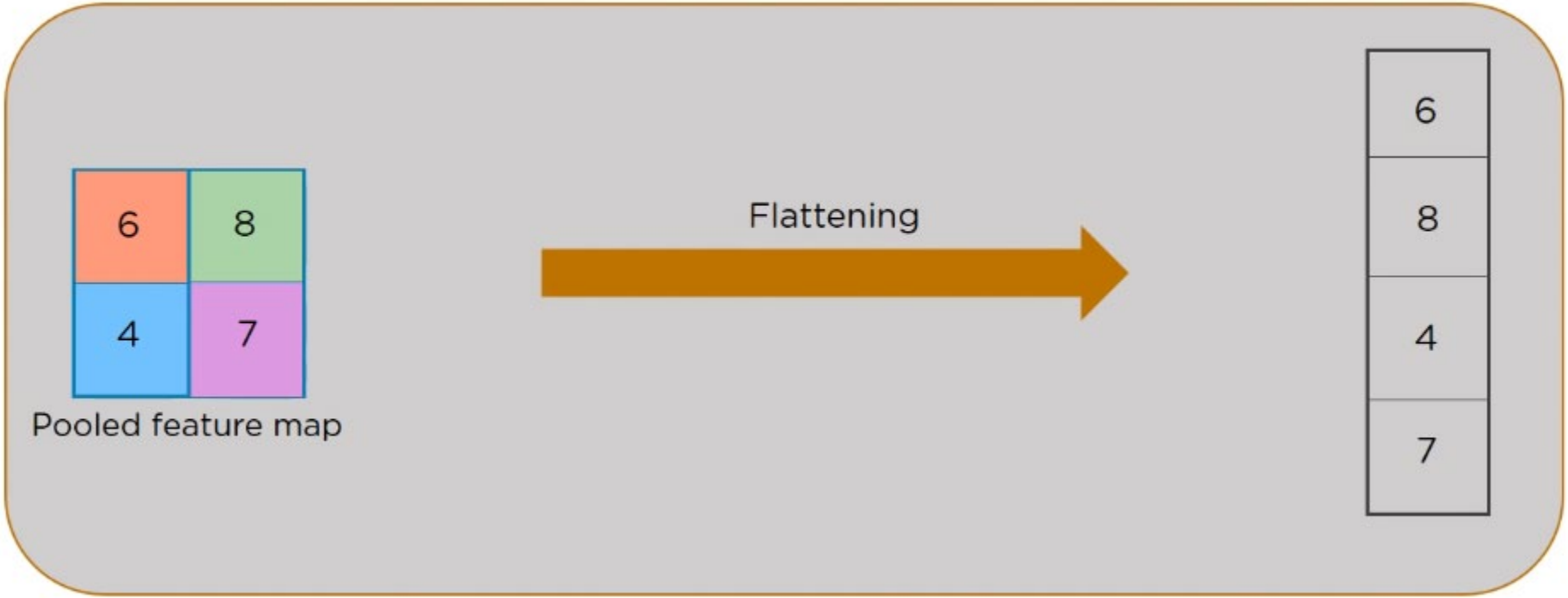
Pooling layer



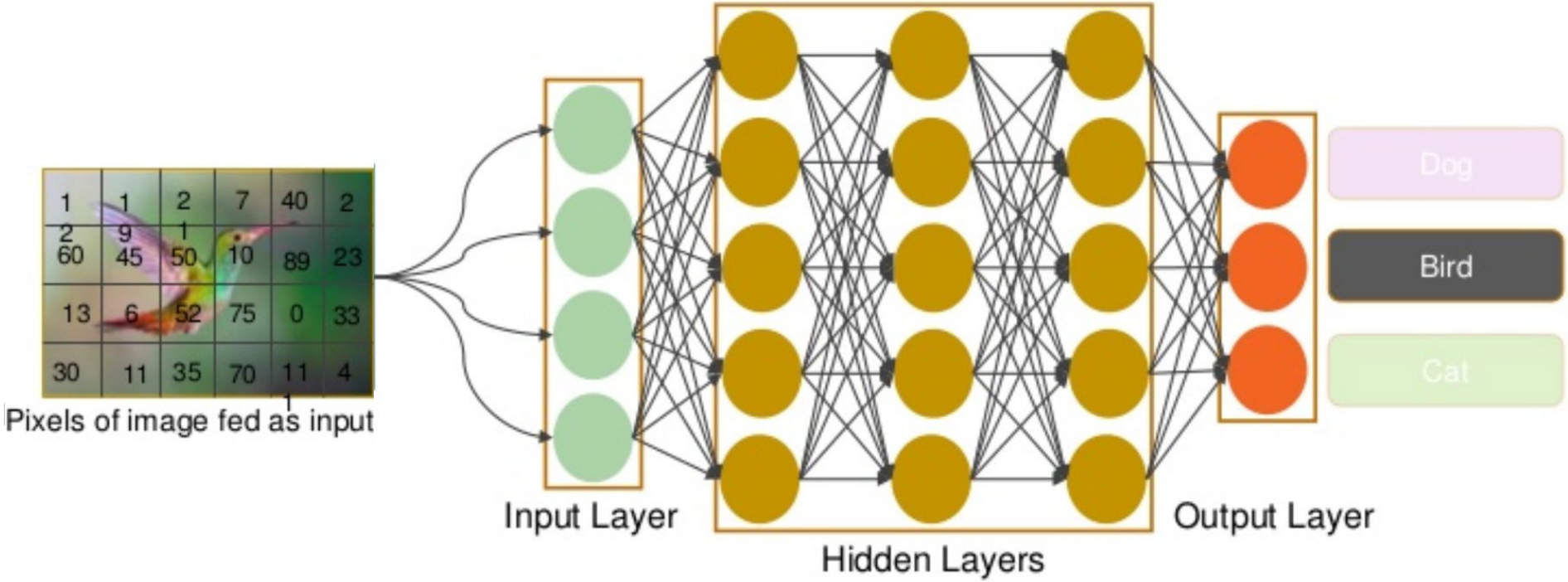
Pooling layer



Pooling layer



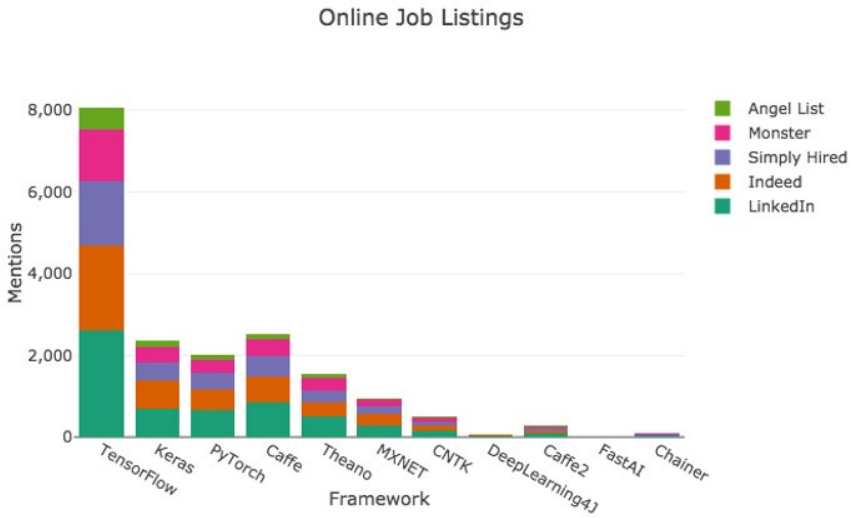
Fully connected layer

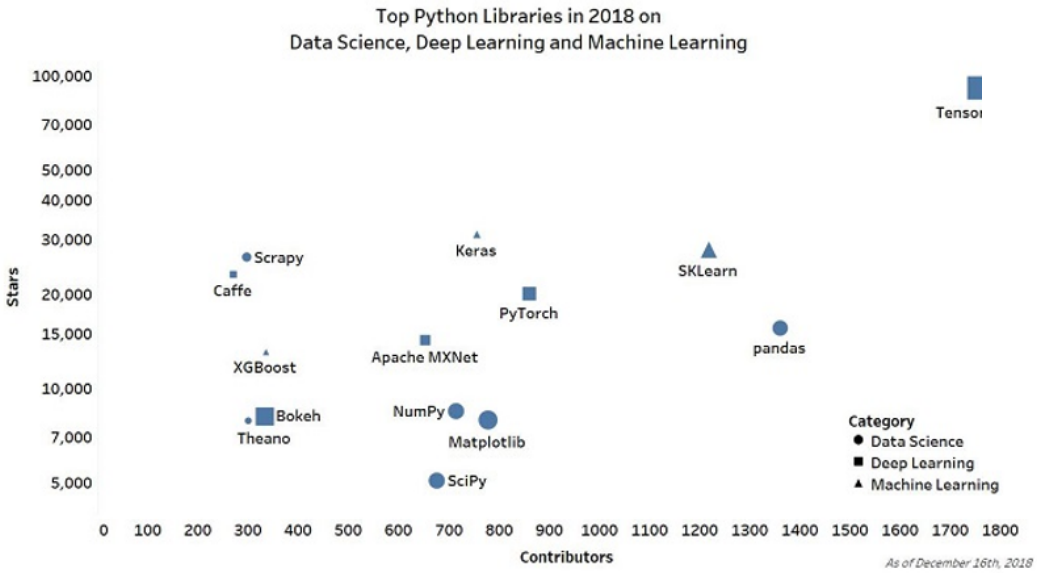


Let's try this out – with TensorFlow

TensorFlow popularit

What is a tensor?





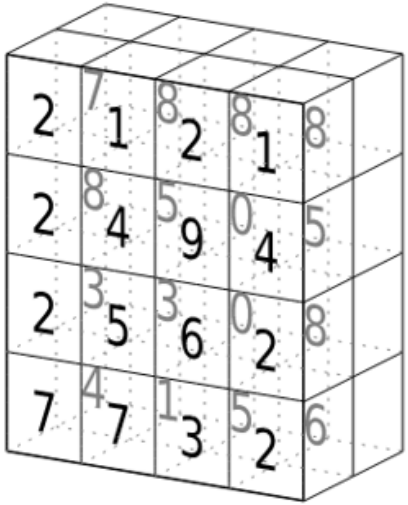
What is a tensor?

't'
'e'
'n'
's'
'o'
'r'

tensor of dimensions [6]
(vector of dimension 6)

3	1	4	1
5	9	2	6
5	3	5	8
9	7	9	3
2	3	8	4
6	2	6	4

tensor of dimensions [6,4]
(matrix 6 by 4)



tensor of dimensions [4,4,2]

TensorFlow on Colab

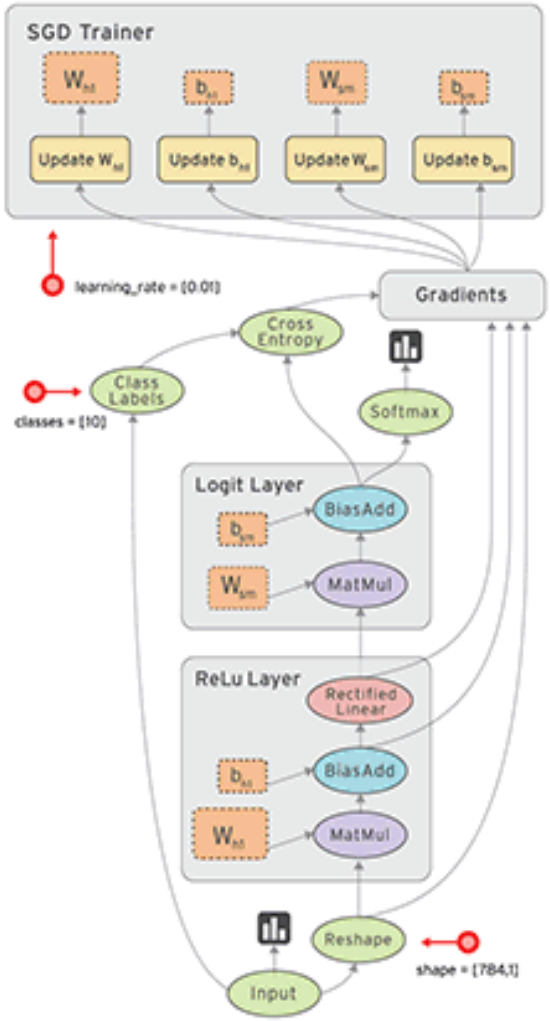
- Enable GPU support
- import tensorflow

TensorFlow tensors

- Constant
  - a = tf.constant(1, name='a_var')
  - b = tf.constant(2, name='b_var')
- Variable
  - c = tf.Variable(a+b)
- Placeholder
  - Define operation $y = ax^2 + bc + c$ using a vector of x values
  - x = tf.placeholder(dtype=tf.float32)

TensorFlow computationa

- An expression of the calculation we want to perform
- with tf.Session() as session:
  session.run(init)
  print(session.run(c))



TensorFlow placeholder

- Define operation $y = ax^2 + bc + c$ using a vector of x values
- x = tf.placeholder(dtype=tf.float32)
- next define the coefficients
  - a = tf.constant(1, dtype=tf.float32)
  - b = tf.constant(-20, dtype=tf.float32)
  - c = tf.constant(-100, dtype=tf.float32)
- next design computational graph with inputs for x
  - y = a * (x ** 2) + b * x + c
  - x_feed = np.linspace(-10, 30, num=10)
- then run it
  - with tf.Session() as sess:
  - results = sess.run(y, feed_dict={x: x_feed})
  - print(results)

TensorFlow operations

Category	Examples
Element-wise Math Ops	Add, Sub, Mul, Div, Exp, Log, Greater, Less, Equal, …
Array Ops	Concat, Slice, Split, Constant, Rank, Shape, Shuffle, …
Matrix Ops	MatMul, MatrixInverse, MatrixDeterminant, …
Stateful Ops	Variable, Assign, …
Neural Network Ops	SoftMax, Sigmoid, ReLU, Conv1D, Conv2D, MaxPool, …
Common Loss Ops	CrossEntropy, L1 / L2 Regularization, RMS, …
Optimization Ops	SGD, Adam, AdaDelta, Adagrad, RMSprop, …
Checkpointing Ops	Save, Restore
Python Language	if, for, while, …

Keras

- API styles
  1. Sequential
  2. Functional
  3. Model subclassing

Our CNN

- MNIST

```
model = keras.Sequential([
    keras.layers.Conv2D(32, kernel_size=(3, 3), activation='relu',
                        input_shape=(28, 28, 1)),
    keras.layers.MaxPooling2D((2, 2)),
    keras.layers.Conv2D(64, kernel_size=(3, 3), activation='relu'),
    keras.layers.MaxPooling2D((2, 2)),
    keras.layers.Conv2D(64, kernel_size=(3, 3), activation='relu'),
    keras.layers.Flatten(),
    keras.layers.Dense(64, activation='relu'),
    keras.layers.Dense(10, activation='softmax')
])
```

Let's try this out