Computer Vision

Computer Vision TOC

Computer Vision TOC (Contd…)

Introduction To Computer Vision

What is computer vision

1. Is a field of study that seeks to enable computing systems to understand and process digital photographs, videos, displays etc. and behave as if they have a vision just as the many living beings have

2. The methods used to achieve this goal, mimic the biological vision mechanism. However, biological vision is still an area of research and is yet to be understood

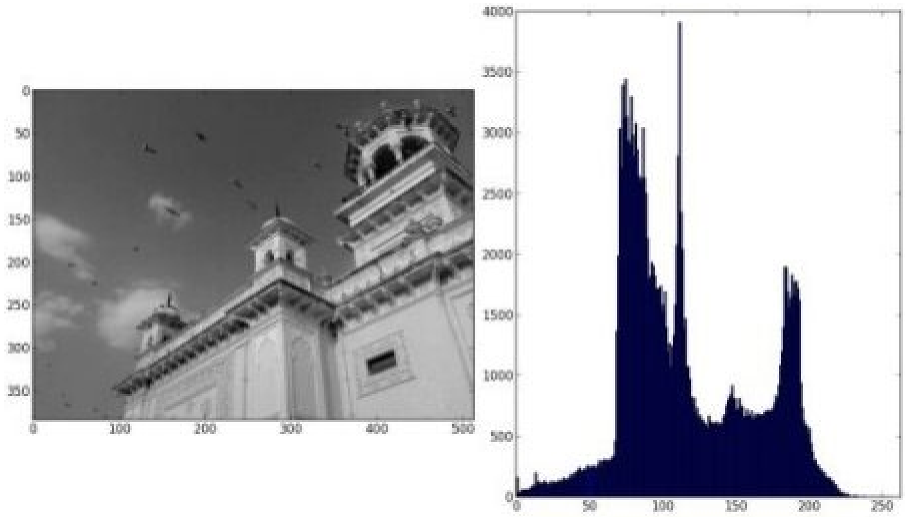
3. “Computer vision” is an overarching term and includes following tasks:

   a. Image classification - label an image based on the object in the image
   b. Object detection / localization / segmentation – Detect an object in an image and localize is using a bounding box.
   c. Segmentation is pixel-wise classification to give the separation of the objects
   d. Similarity learning – Which two images are similar based on the content of the image
   e. Image captioning – Describing the image (combines NLP with computer vision)
   f. Generative modelling – Generate images based on the style of another image
   g. Video analysis – process the entire set of digital frames for object detection and tracking

Approaches to Computer Vision

Pixel intensity histograms

1. Analyze digital images based on pixel intensity histograms. This histogram is a graph showing the number of pixels in an image at each different intensity value found in that image

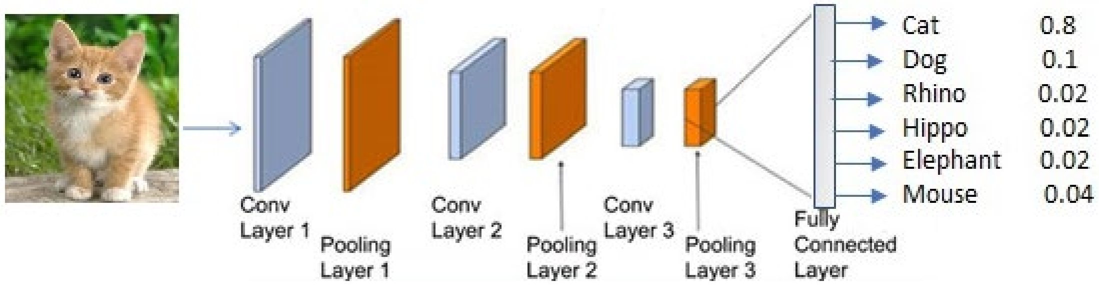


Source: CV - home.jpg

2. Two pictures with similar pixel intensity histograms are likely to be similar!

Convolutional Neural Networks (CNN)

1. Is a class of deep neural networks, most commonly used in image analysis. It is the default method for all the listed computer vision objectives



2. CNN is based on the mathematical concept of convolution though it is not exactly the same

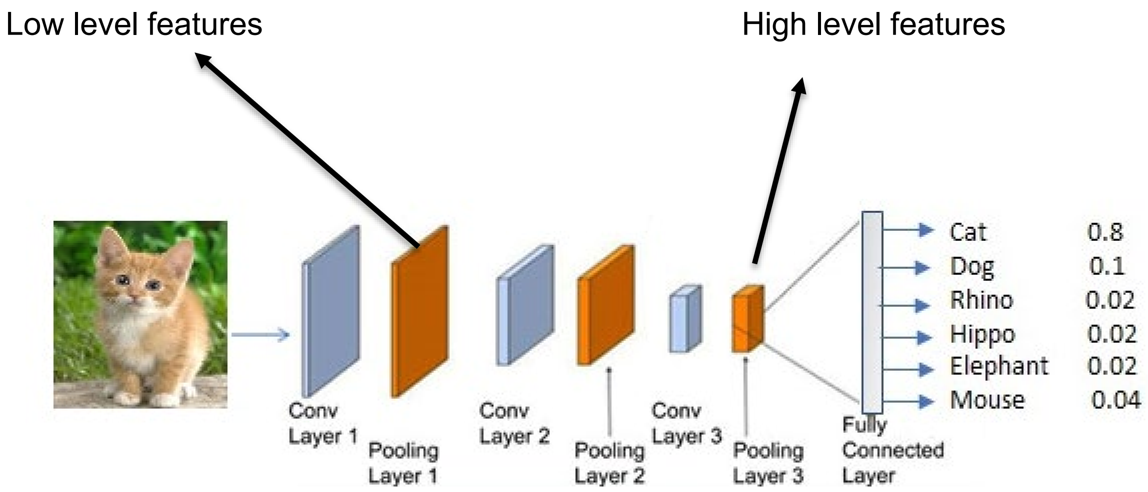
3. Through successive convolutions and pooling (a.k.a. subsampling), the technique identifies the features of the input object

4. The features extracted are used by the fully connected dense layer along with the corresponding label to train the entire network for image classification and other purposes

Convolutional Neural Networks (CNN)

5. CNNs typically work on pixel intensity value changes and learn to process them in a way that makes it possible to accomplish a certain computer vision task, such as image recognition.

6. Internal layers of CNNs can be considered as image filters that extract information on different hierarchy levels where higher hierarchy reflects features on large scale. Using these information they accomplish their tasks



Digital Images and Pixels

Analog to Digital Image

1. A digital image d[m,n] described in a 2D discrete space (integral values)

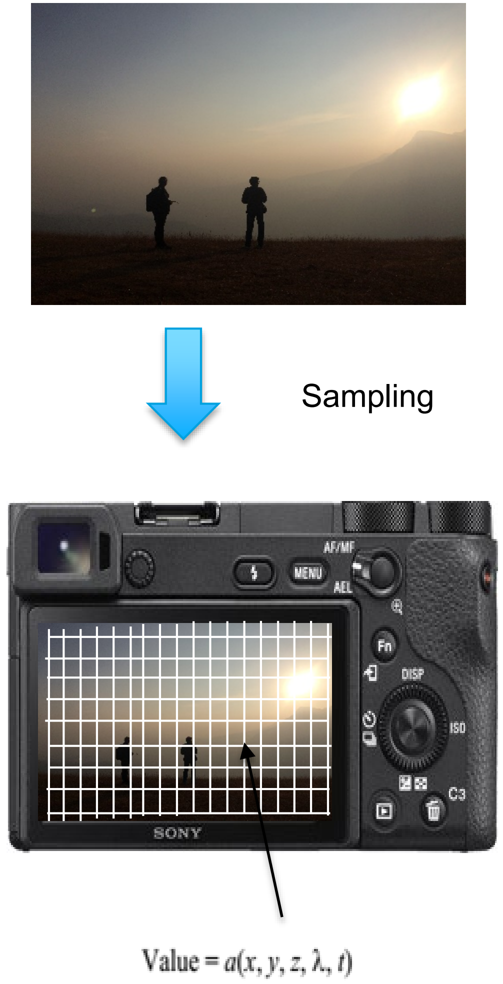
2. It is derived from an analog image a(x,y) in a 2D continuous space through a sampling (shutter open and close) a.k.a. digitization

3. The 2D analog image a(x, y) is divided into N rows and M columns. The intersection of a row and a column is called a pixel.

4. The value assigned to the integer coordinate of a digital image d[m,n] is a function of characteristics of real signal impinging on the sensor in that coordinate

5. The value assigned to every pixel is the average brightness (varies during the sampling time) in the pixel rounded to the nearest integer value

6. The process of representing the amplitude of the 2D signal at a given coordinate as an integer value is usually referred to as **amplitude quantization**

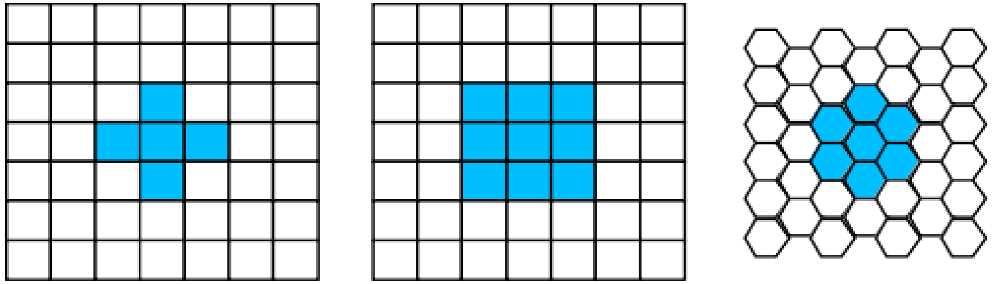


Digital Image and Neighborhood

1. Digital image processing in CNN will involve generating new image from existing digital image using a transformation process

2. The values at a pixel in the new image will be dependent on the input values in the neighborhood of the same pixel on the original image. A.k.a localized operations

3. Neighborhood types – defines the way neighbors are identified in the image-
   a. Rectangular sampling – Images are sampled laying rectangular grid on the over source image
   b. Hexagonal sampling – Images are sampled laying hexagonal grid over the source image
   c. In CNN, we will restrict to rectangular grids due to software limitation



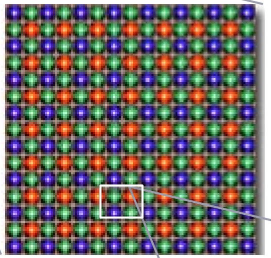
Rectangular sampling 4-connected | Rectangular sampling 8-connected | Hexagonal sampling 6-connected

Digitization - Analog to Digital Images

Pixels – Light wells

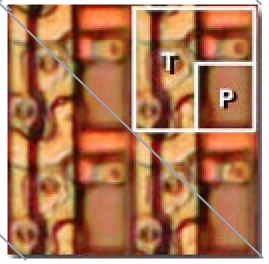
Image source :
http://olympus.magnet.fsu.edu/primer/digitalimaging/cmosimagesensors.html

Light sensors behind camera objective lens



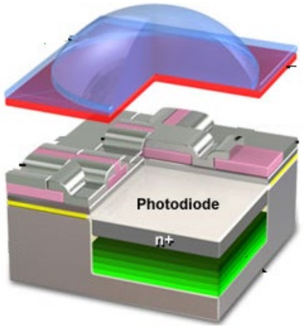
Bayer color filter grid
Ngreen = Nred +Nblue

Color filter acts like a micro lens to focus light on the photo detector below



A 2X2 grid of photo detectors under each micro lens. The P part generate electrons. The T is other supporting unit

Anatomy of the pixel



Pixels – Light wells

1. Light is made of photons, smallest energy packets. They carry information from the source
2. On collision with certain types of metals, they produce electrons (photoelectric effect)
3. When they hit the CMOS image sensors in the camera behind the lens, electrons are released from the sensors producing a small electric current.
4. The sensor is a grid of light sensitive pixels and each pixel gathers the free electrons
5. Each pixel well has a maximum capacity of electrons it can collect. This maximum is known as full well capacity.
6. The color intensity of each pixel is determined by the amount & kind of light information it collects

greatlearning
Learning for Life

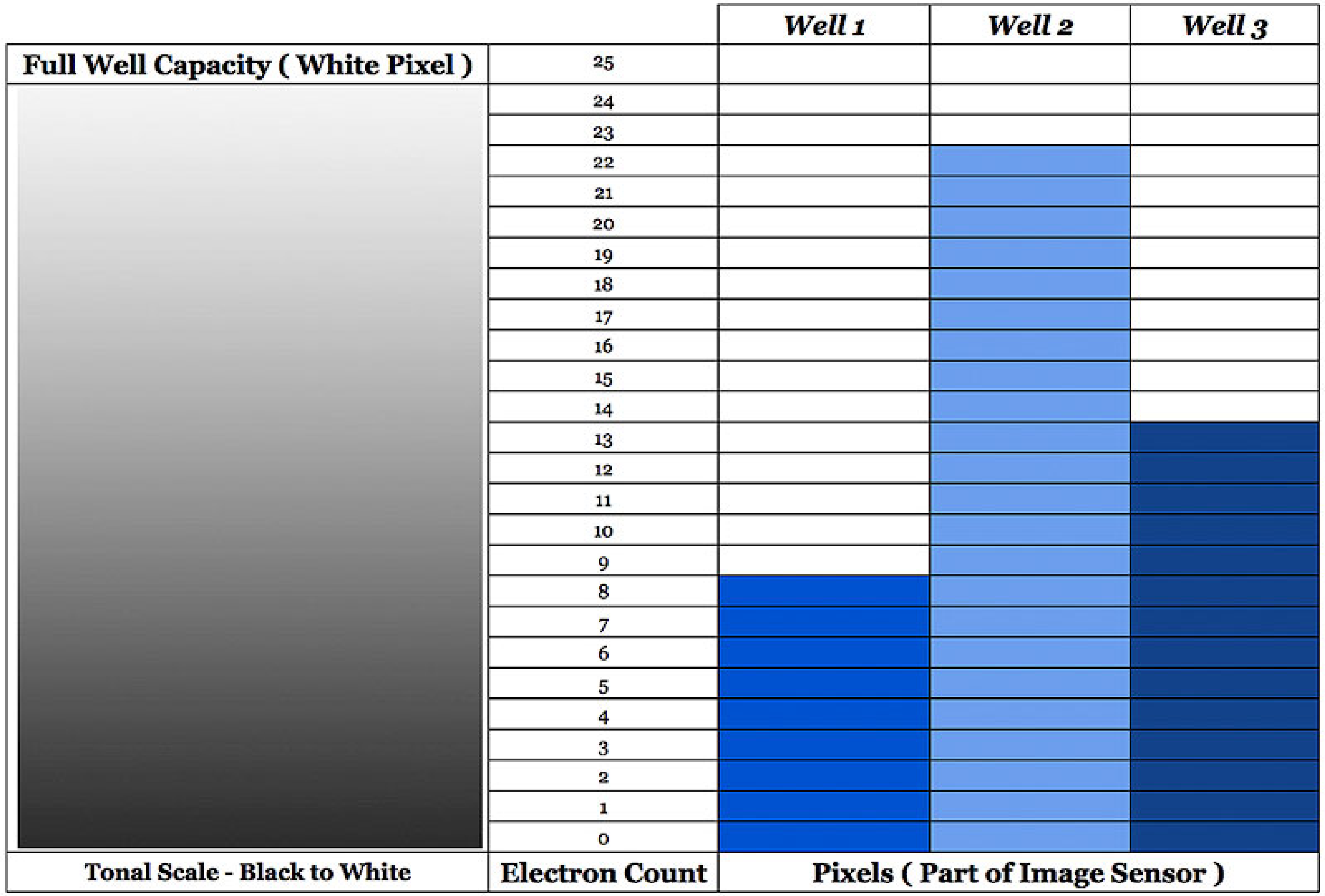
Pixels – Light wells

7. On completion of the sampling window (shutter close), the electrons collected in each pixel cell is converted to a digital signal representing color and intensity (R,G,B in the range of 0 to 255)
8. A white pixel contains the maximum amount of electrons
9. A black pixel contains no electrons

greatlearning
Learning for Life

Pixels – Light wells

The closer the number of electrons collected to full capacity, the lighter the pixel gets



Digital Image Sizing

Digital Image Sizing

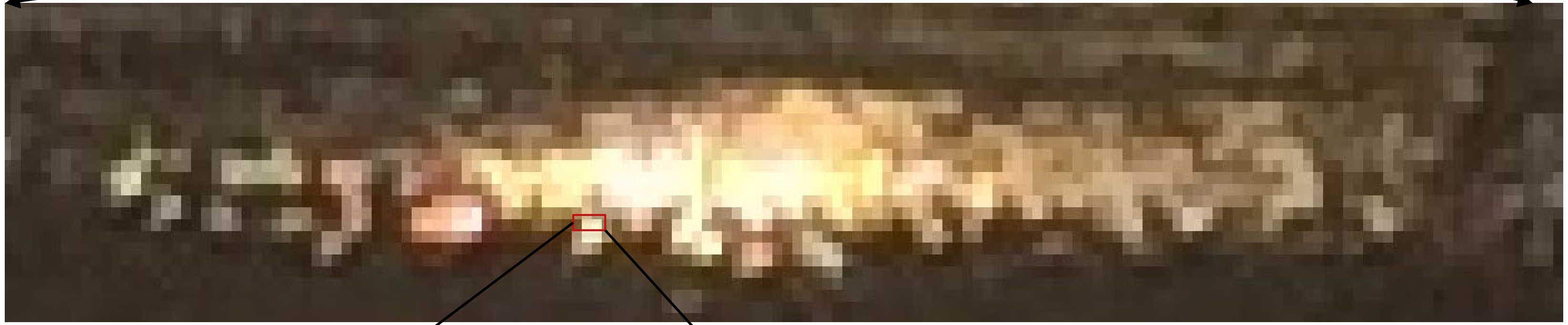
Analog Image

Sampling

Digitization

Digital Image

Pixels for a small section of the digital image



11011110, 10111000, 10000111
R G B

Single pixel stored in bytes in Red, Green, Blue

Digital Image Sizing

Imagine the picture was 5X5 pixel in size

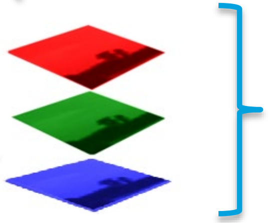
Each pixel has R, G, B layers

Each layer i.e. (R, G, B) can take 256 values from 0 – 255 representing 256 different shades respectively

We need 8 bits / 1 byte to store 256 values because $2^8 = 256$

Therefore we need 3 bytes (for R, G, B layers respectively) for every pixel

Thus, this 5X5 color image will need 25 X 3 = 75 bytes



Each pixel (a rectangle in this example) is made out of combination of different shades of Red, Green and Blue

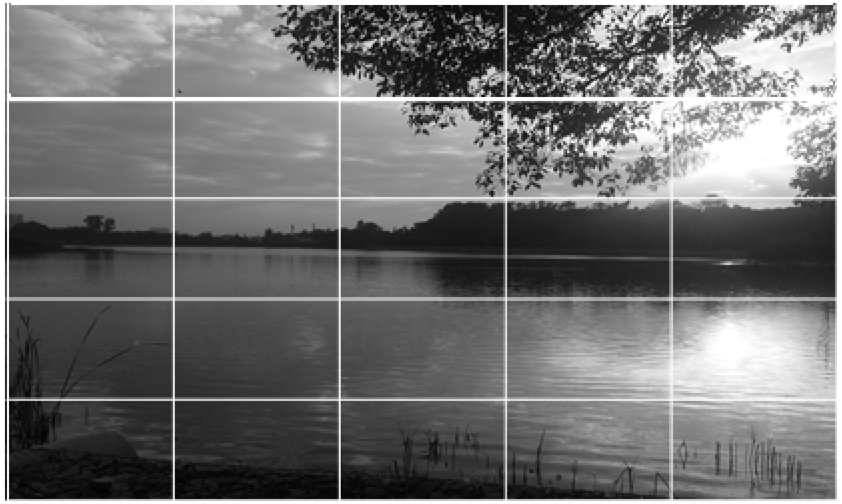
Digital Image Sizing

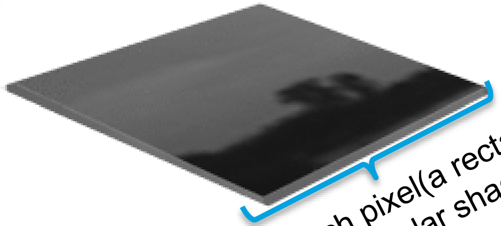
If the 5X5 image was a grey scale image then each pixel can take colors black to white with different shades

There are 256 shades of black where 0 is absolute black and 255 is absolute white

Each pixel has only one layer of grey scale which means each pixel needs only 8 bits / 1 byte to hold its shade of black

Thus, this 5X5 color image will need 25 X 1 = 25 bytes





Each pixel(a rectangle) is made of a particular shade of grey.

Note : in this picture each rectangle is made of different shades because it is not really a pixel

greatlearning
Learning for Life

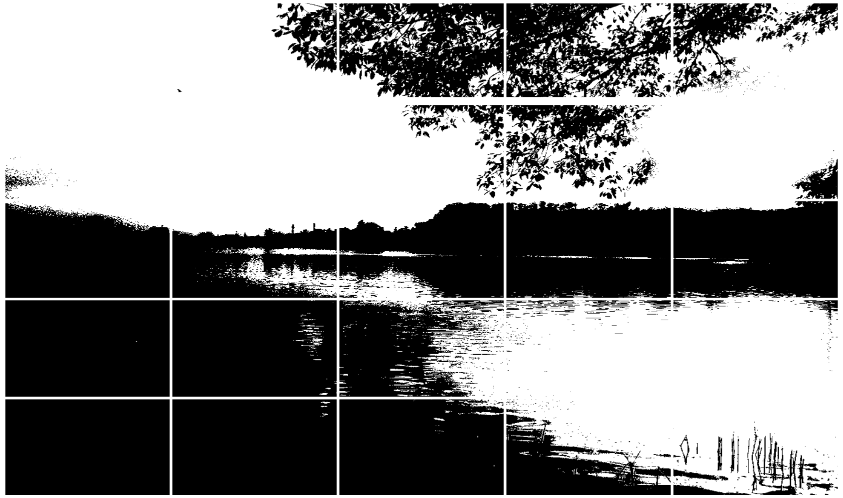
Digital Image Sizing

If the 5X5 image was a black and white only image

Each pixel can be black or white and nothing in between

Then each pixel needs only 1 bit (0 – black, 1 – white)

Thus, this 5X5 color image will need 25 X 1 = 25 bits



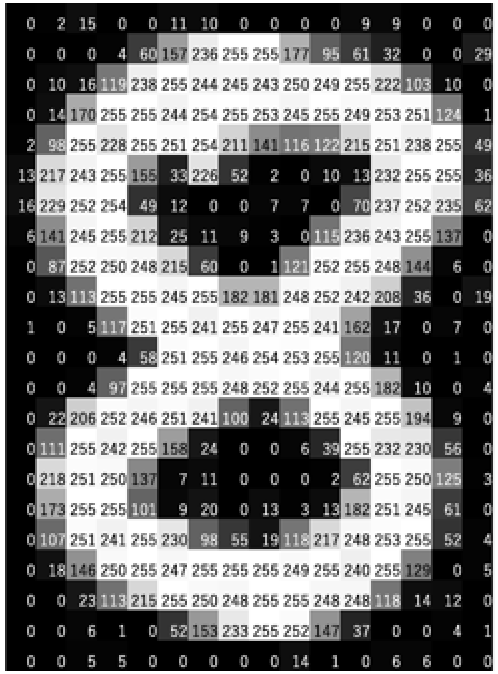
Digital Image Sizing

To summarize relation between number of bits per pixel (pixel depth), number of colors, image size (rows and columns of pixel

bit-depth	max colors	file size of 300x500 image
1	2	18kB
2	4	37kB
3	8	55kB
4	16	73kB
5	32	91kB
6	64	110kB
7	128	128kB
8	256	147kB

Image as a grid of discrete integral numbers

Images as seen by a computer



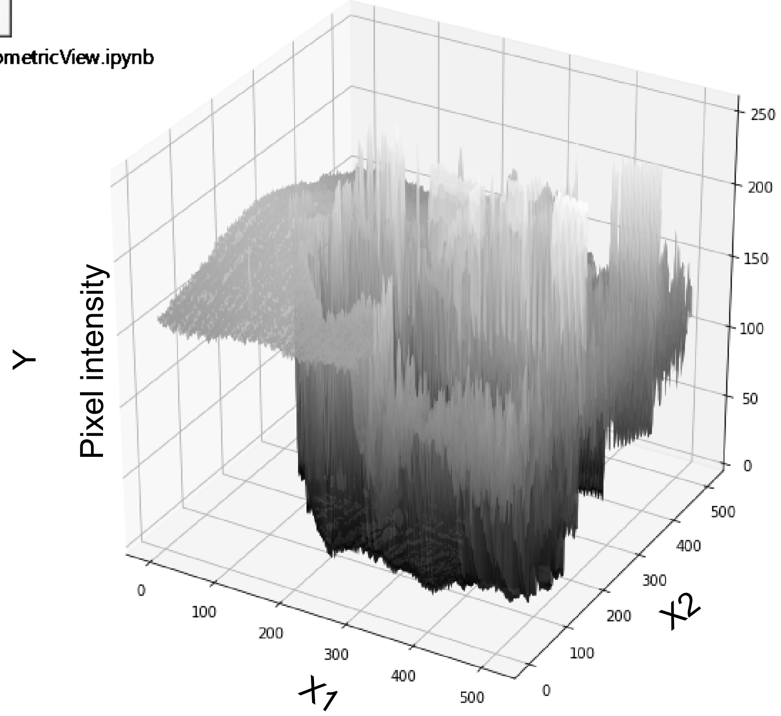
0	2	15	0	0	11	10	0	0	0	0	9	9	0	0	0
0	0	0	4	60	157	236	255	255	177	95	61	32	0	0	29
0	10	16	119	238	255	244	245	243	250	249	255	222	103	10	0
0	14	170	255	255	244	254	255	253	245	255	249	253	251	124	1
2	98	255	228	255	251	254	211	141	116	122	215	251	238	255	49
13	217	243	255	155	33	226	52	2	0	10	13	232	255	255	36
16	229	252	254	49	12	0	0	7	7	0	70	237	252	235	62
6	141	245	255	212	25	11	9	3	0	115	236	243	255	137	0
0	87	252	250	248	215	60	0	1	121	252	255	248	144	6	0
0	13	113	255	255	245	255	182	181	248	252	242	208	36	0	19
1	0	5	117	251	255	241	255	247	255	241	162	17	0	7	0
0	0	0	4	58	251	255	246	254	253	255	120	11	0	1	0
0	0	4	97	255	255	255	248	252	255	244	255	182	10	0	4
0	22	206	252	246	251	241	100	24	113	255	245	255	194	9	0
0	111	255	242	255	158	24	0	0	6	39	255	232	230	56	0
0	218	251	250	137	7	11	0	0	0	2	62	255	250	125	3
0	173	255	255	101	9	20	0	13	3	13	182	251	245	61	0
0	107	251	241	255	230	98	55	19	118	217	248	253	255	52	4
0	18	146	250	255	247	255	255	255	249	255	240	255	129	0	5
0	0	23	113	215	255	250	248	255	255	248	248	118	14	12	0
0	0	6	1	0	52	153	233	255	252	147	37	0	0	4	1
0	0	5	5	0	0	0	0	0	14	1	0	6	6	0	0

Source : http://yann.lecun.com/exdb/mnist

Image as a function

Image as a function

GreyScaleImage_IsometricView.ipynb



1. Take any image from skimage library
2. Greyscale the image
3. Let us look at it in isometric view (for ease of understanding next step)
4. Convert the greyscale image into a 3D plot
5. Z axis is magnitude of pixel intensity (0 – black, 255 white)
6. The 2D greyscale in 3D plot of pixel intensity vs x, y position
7. This 3D surface is a function representing the picture as a mapping of x,y coordinate to pixel intensity

Edges as features

1. Can you read this line written here => . Obviously you could not read that. But you are able to read this line… Why?
2. When the color of an object is same as background, you cannot see it (Ref: - https://tenor.com/view/octopus-camo-boo-watchingme-hiding-gif-11776143
3. Hence, for us to be able to see an object, the object’s color and associated properties must be different from the background
4. When so, the object gets a distinct boundary at overall level. Further, within the boundary of the object may be sub-boundaries marking separate regions such as eye
5. It is these boundaries that act as features in object detection and recognition. The main motive for computer vision
6. Boundaries, also known as edges are those positions in space where the characteristics of light change
7. Thus, detecting the changes in the light helps detect edge and edge detection is the key to computer vision

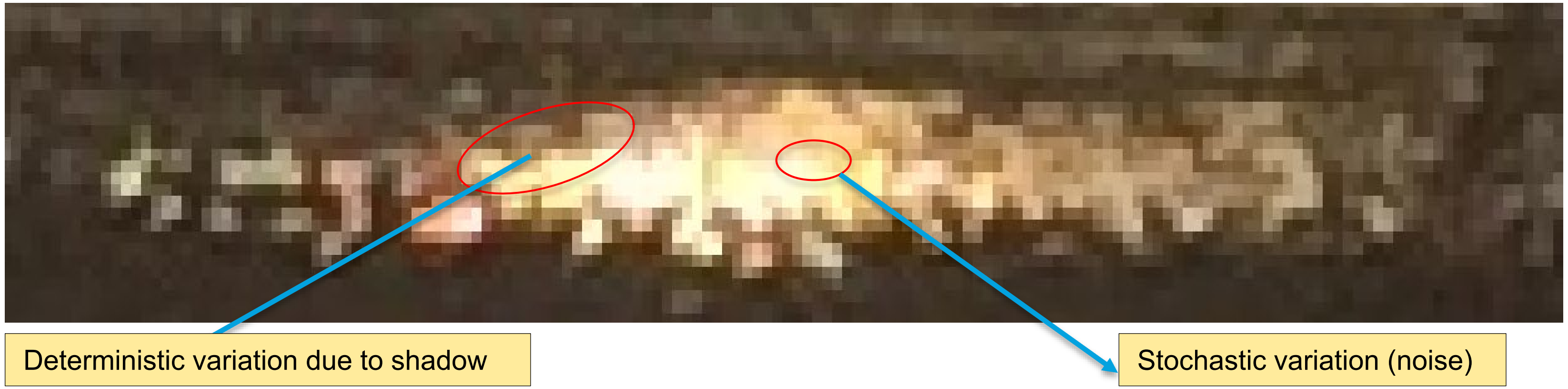
Noisy Edge! / Digital Noise

1. When working at pixel level, the edges that appear as change in pixel color or intensity, can be due to the true edge of an object but may also be caused by the digital noise
2. Digital noise due to dark current can artificially create edges where there are none. If these noisy edges are not taken care of, the CNN algorithm can misinterpret them as genuine edges
3. This is similar to noisy features that we come across in conventional machine learning

Sources of digital noise

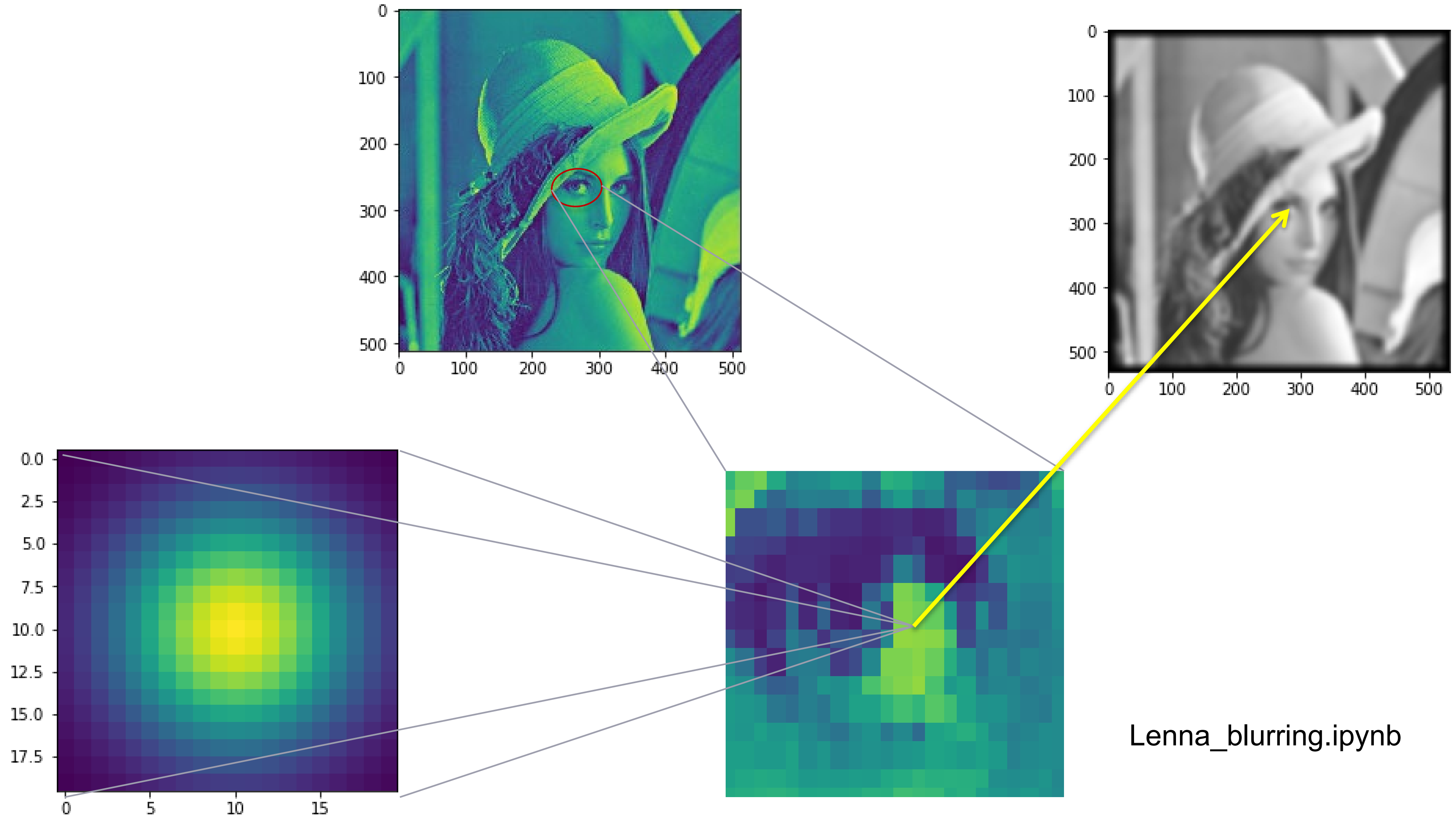
1. Digital images are usually contaminated by a variety of noise generators that act while the digital image is being produced from the analog

2. Digital noise refers to the stochastic variations in the pixel values against the actual variance in the analog signal



3. Sources of noise –
    1. Photon noise due to quantum nature of light. Modern sensors can capture individual photons but the generation of photon is probabilistic (quantum mechanics)
    2. Thermal noise due to heating of the devices. The heat kicks out additional electrons from the sensors along with the electrons generated by photoelectric effect. A.k.a dark current
    3. On-chip electronic noise created by fields surrounding the devices such as FET (Field Effect Transistors , amplifier noise etc..)

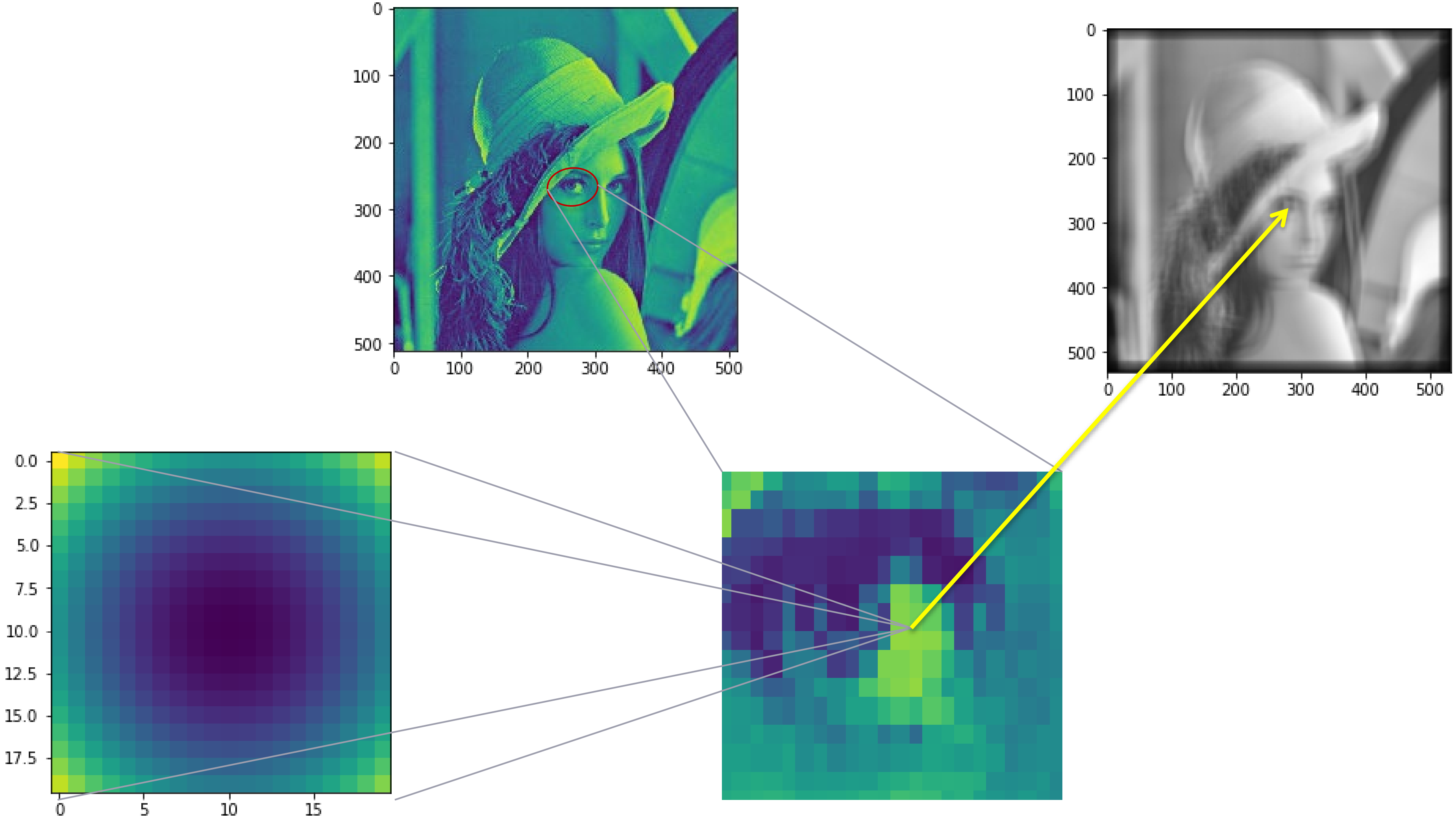
Addressing digital noise through blurring / Gaussian blurring



Lenna_blurring.ipynb

greatlearning
Learning for Life

Addressing digital noise through blurring / Gaussian blurring



greatlearning
Learning for Life

Image processing (Gaussian Blurring)

Why would anybody in their right mind apply blurring filters to their image?

1. Every image is associated with noise. Noise is the pixel brightening because of the dark current i.e. the current generated not by the photons but other mechanisms

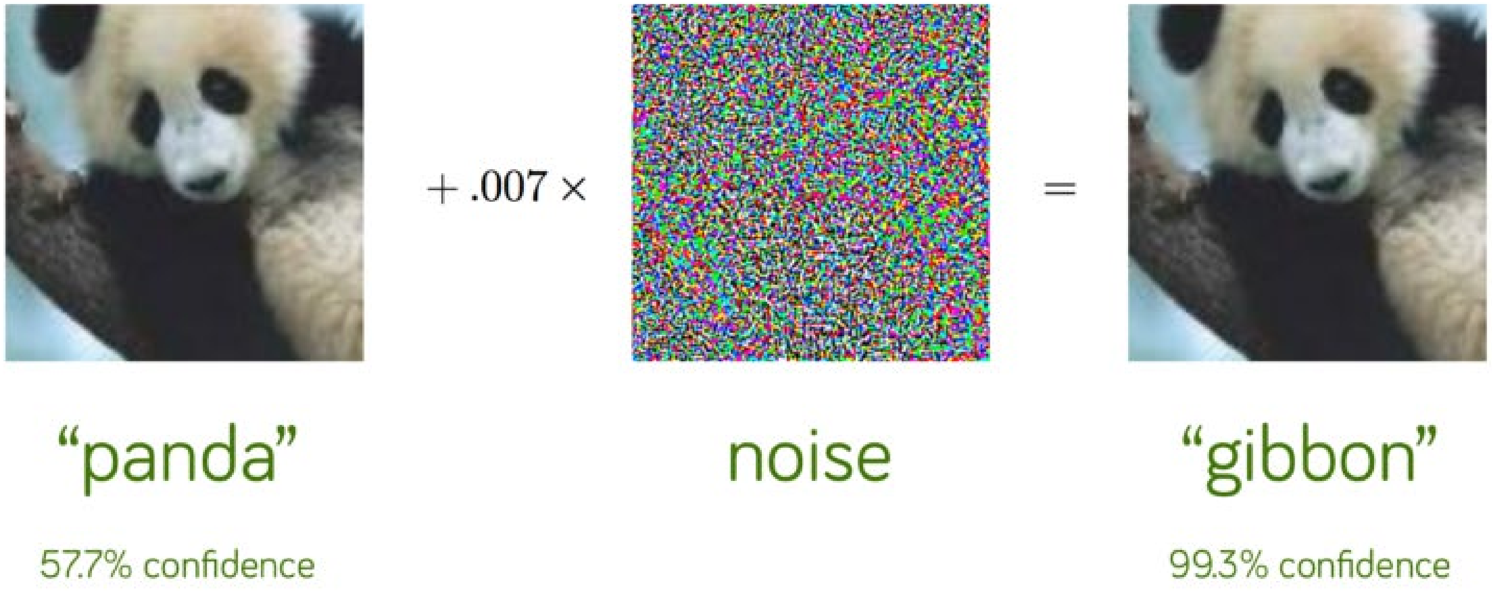
2. These unwanted specks of brightness can hamper the processing down the line

3. Like in any conventional algorithms, we do not want our models to get influenced by noise!

4. The gaussian blurring and other filters act like erasers on the noisy pixels by re-calculating their pixel values based on neighbor pixels!

5. Hopefully, after this step, the digital image will have only the object of interest (the real signal or function) that we wish to work on

Image processing (Gaussian Blurring)

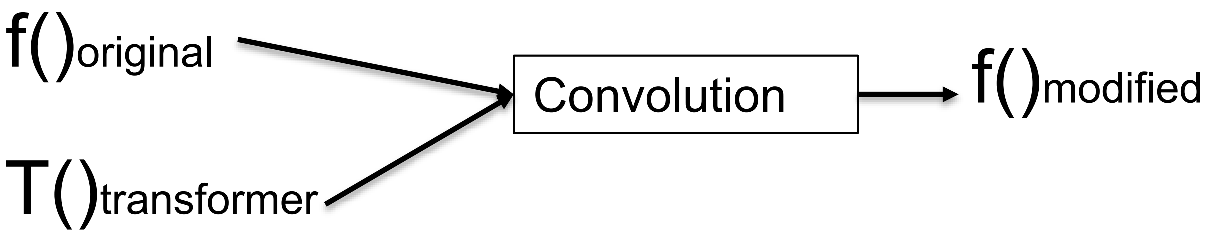


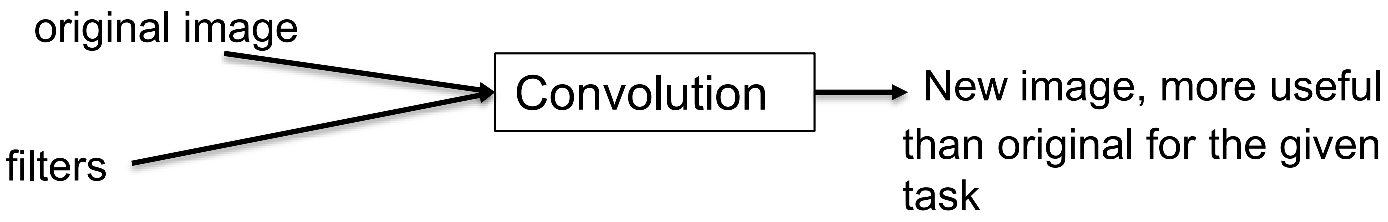
https://www.kdnuggets.com/2019/03/breaking-neural-networks-adversarial-attacks.html

Convolutional Neural Networks

Convolution for image processing

Convolution is a mathematical transformation of a given function (analog or digital) into a from that is more useful than the original function given a requirement for e.g. image classification.





Pixel Intensity Gradient

1. Through the use of a filter function on the image function (convolution) , we try to detect edges.

2. The edges appear as a gradient at a pixel (is the direction in which intensity changes maximum in that pixel's neighborhood

3. The gradient is searched in both X and Y direction and a consolidated gradient is found using vector addition (gradients are vectors)

4. The larger the magnitude of the gradient, the stronger the evidence of presence of an edge / a feature

Convolution for Edge Detection

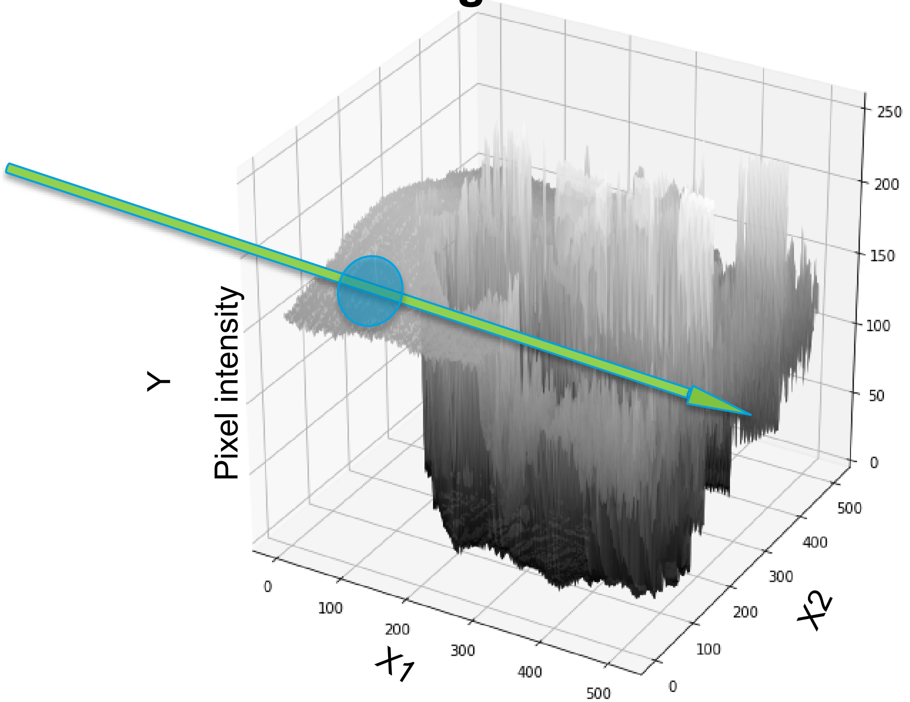
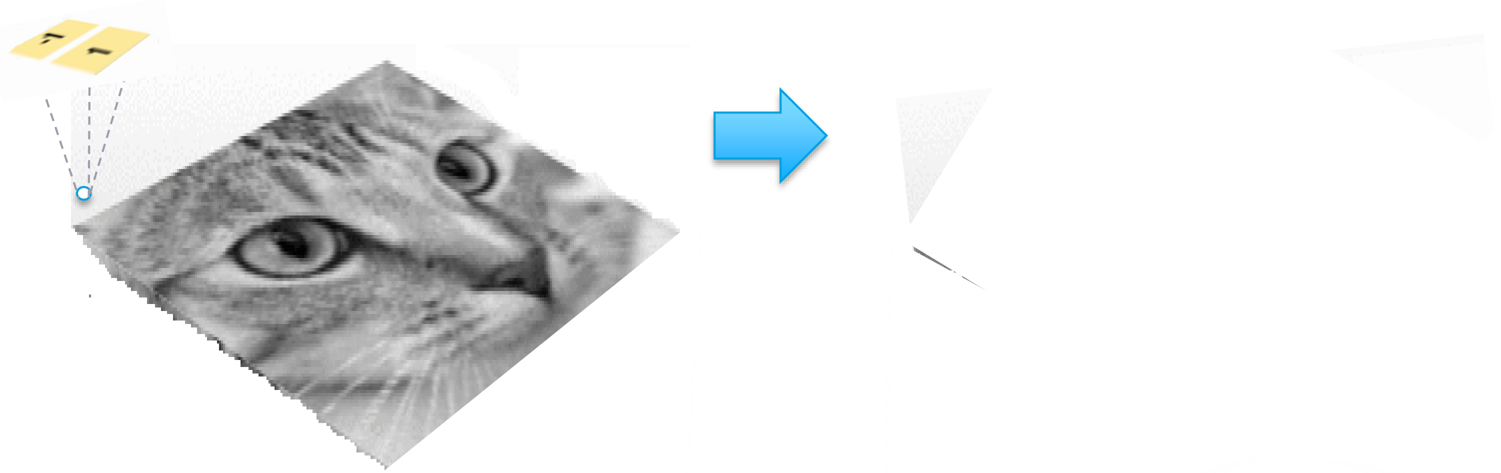


Image Gradient –

1. The raw pixels, with their intensity across x, y is not very useful
2. Difference in intensity of neighboring pixels contains richer information
3. Difference in pixel intensity across one dimension is shown as slope (green line) this dy /dx1. Since dx1 is going to be 1 pixel, the slope is just dy/1 = dy
4. dy is difference in intensity of a pixel and it's previous pixel! i.e. y2 – y1
5. The function dy/dx is a kernel function that acts on input image function, gives slope at every pixel
6. The kernel is implemented using a matrix shown in yellow sliding over the image
7. The process of scanning the whole 2D space using the filter is known as convolution (note: the convolution in CNN is not exactly the same convolution we come across in mathematics)

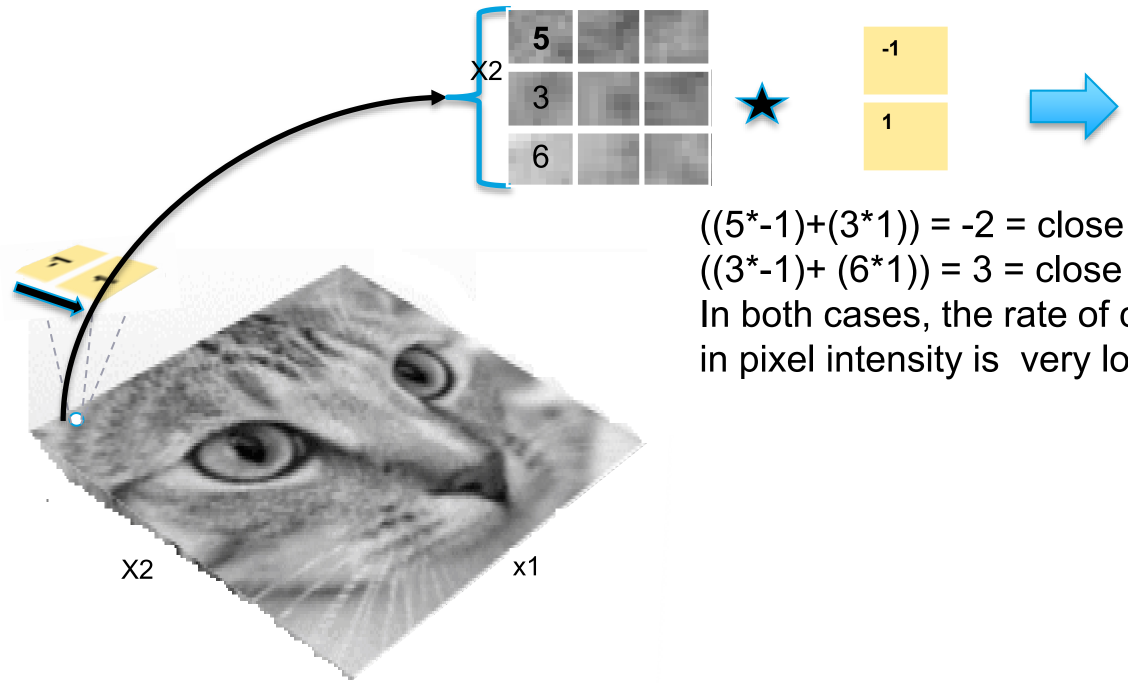
Convolution for Edge Detection / Horizontal edge detection

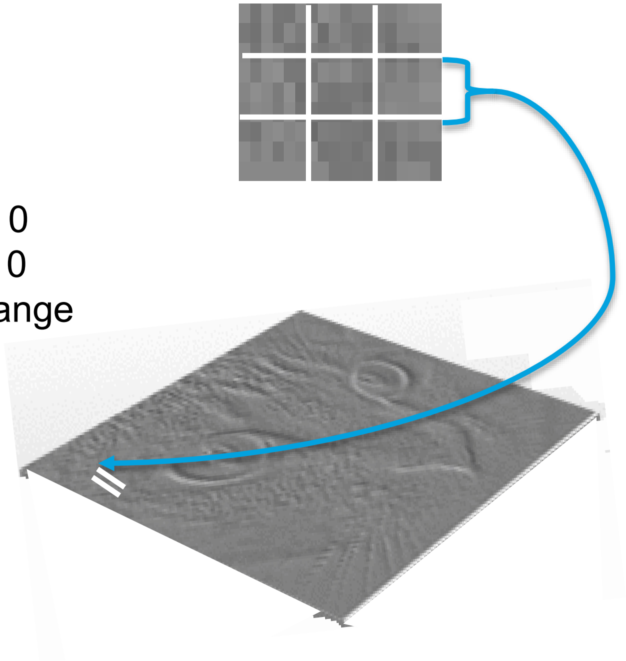


A vertical movement of the filter will give horizontal edges
A horizontal movement of the filter gives vertical edges (not shown above)
Such simple filters subtract next pixel value from current to get the difference and may not always give good results (as in the case of camera man image)
Instead we use a different filter such as the Sobel filter which detects edges well

greatlearning
Learning for Life

Convolution for Edge Detection / Horizontal edge detection





((5*-1)+(3*1)) = -2 = close to 0
((3*-1)+ (6*1)) = 3 = close to 0
In both cases, the rate of change in pixel intensity is very low.

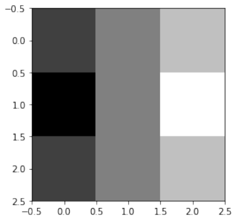
1. Applying filter (yellow) vertically on a cat image
2. A 2X1 grid of pixels convolutes with the pixel grid with respective pixel values
3. The operation is equal to dy/dx2 where dy is change in pixel value and dx2 one unit jump in row i.e. 1
4. When dy is close to 0 the pixels in that neighborhood are similar in intensity i.e. not very different
5. Result close to 0 indicates a homogeneous neighborhood / no interesting pattern or info there
6. When we come to the eybrows, the magnitude of the result of convolution will be high and higher numbers will be close to white. This appear as white edge in the result (eye part)

Type of Filters

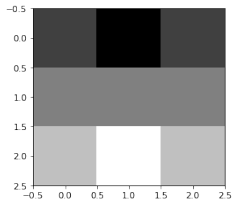
Sobel operator is defined for two directions x1, y and they approximate the gradient at a point on the given image

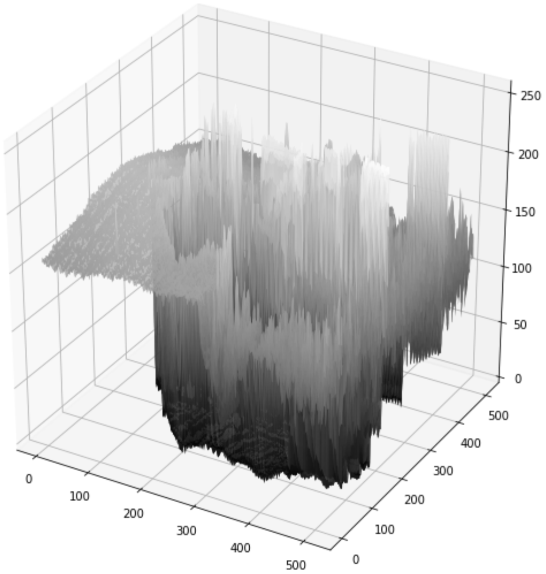
Gx1 is result of convolution of the image with the Sobel operator in x1 direction and Gy is the result of convolution of the image with the Sobel operator in y direction

```
Hx gradient
 [[-1.  0.  1.]
 [-2.  0.  2.]
 [-1.  0.  1.]]
```



```
Hy gradient
 [[-1. -2. -1.]
 [ 0.  0.  0.]
 [ 1.  2.  1.]]
```

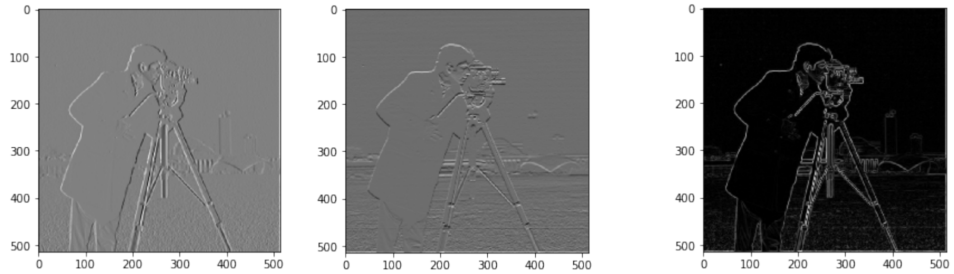




The job of the arrow in the animation earlier, is done by the filters here

Image gradient using Sobel operator / filter

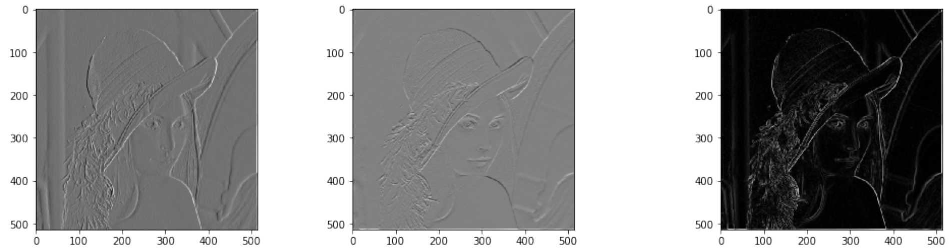
After finding the horizontal and vertical edges, we take the Euclidian based total gradient ie vector addition of the gradient in x and y direction together



Vertical Edge Detection

Horizontal edge detection

Combined result



ANN Vs CNN for Digital Image Processing

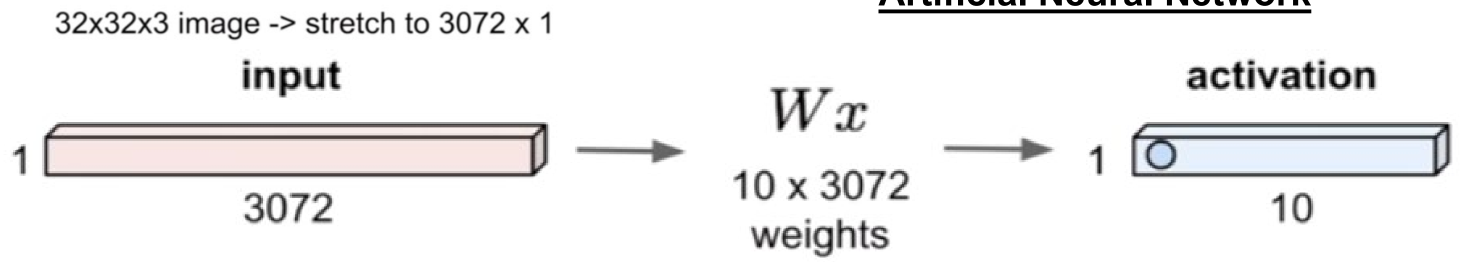
ANN and CNN

1. CNNs, like artificial neural networks, are made up of layers of neurons
2. Each neuron is associated with learnable weights and biases.
3. Each neuron receives several inputs, takes a weighted sum over them
4. The weighted sum is passed through a non-linear activation function to generate a response
5. The whole network has a loss function that is optimized using back-prop algorithm
6. But Convolutional neural networks operate on tensors (volumes)

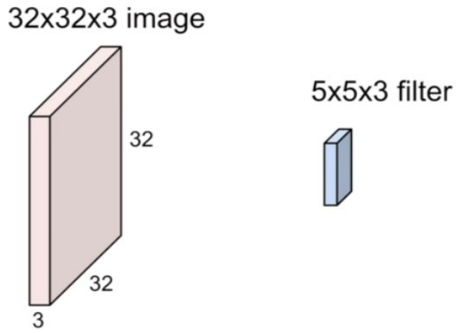
ANN Vs CNN

1. Artificial neural network takes a vector as input. Spatial structure is ignored

Artificial Neural Network



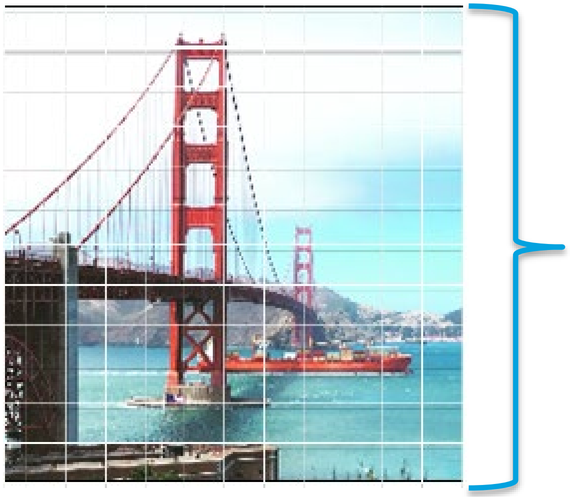
2. Unlike ANN which work on vectors, CNN works on volumes maintaining spatial structures



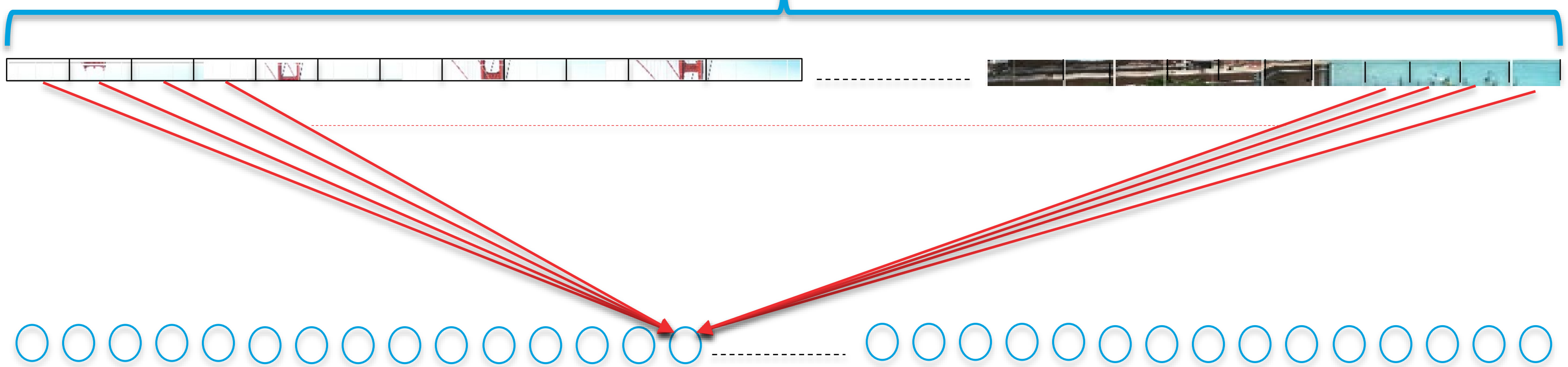
Convolutional Neural Network

Filter will slide over the image (convolve)
computing dot product
A single filter will apply on all 3 color channels

Images and First Hidden Layer in ANN

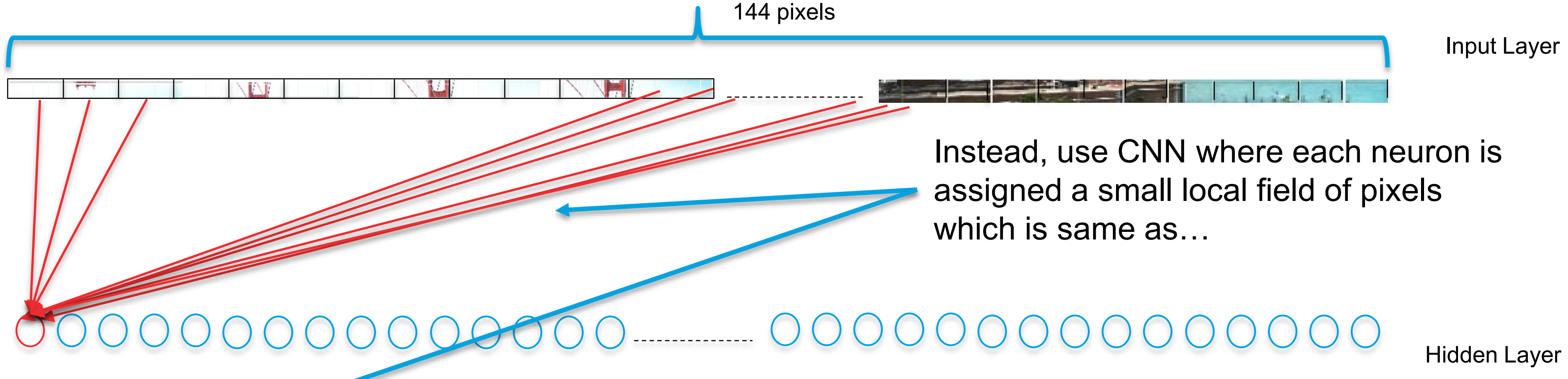


Using fully connected neural network for image processing will lead to huge number of coefficients to learn in the first hidden layer

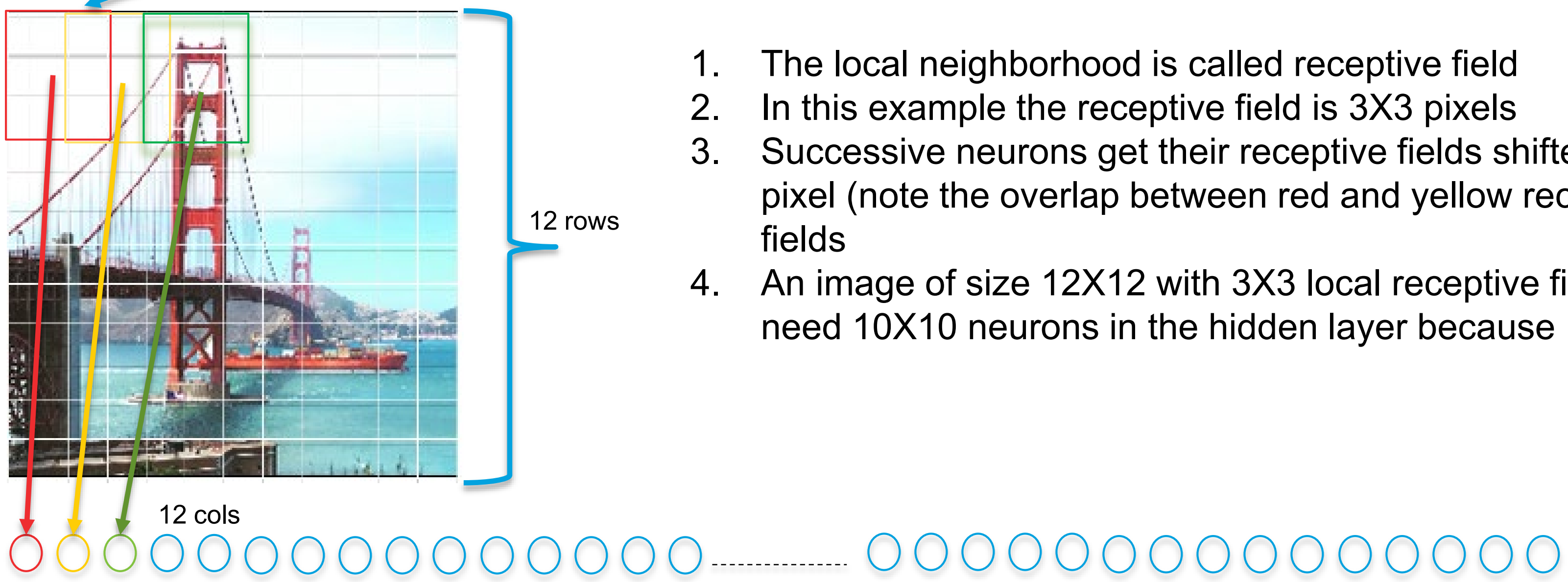


One neuron will have 144 weights to learn
The hidden layer will have 144 X number of neurons to learn

Images and First Hidden Layer in ANN

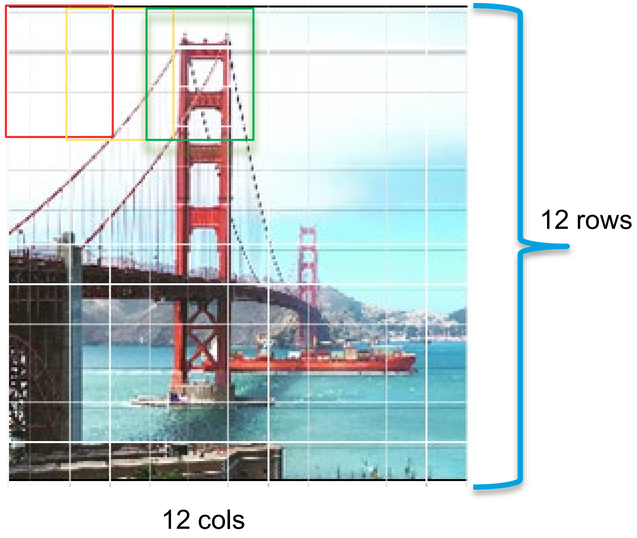


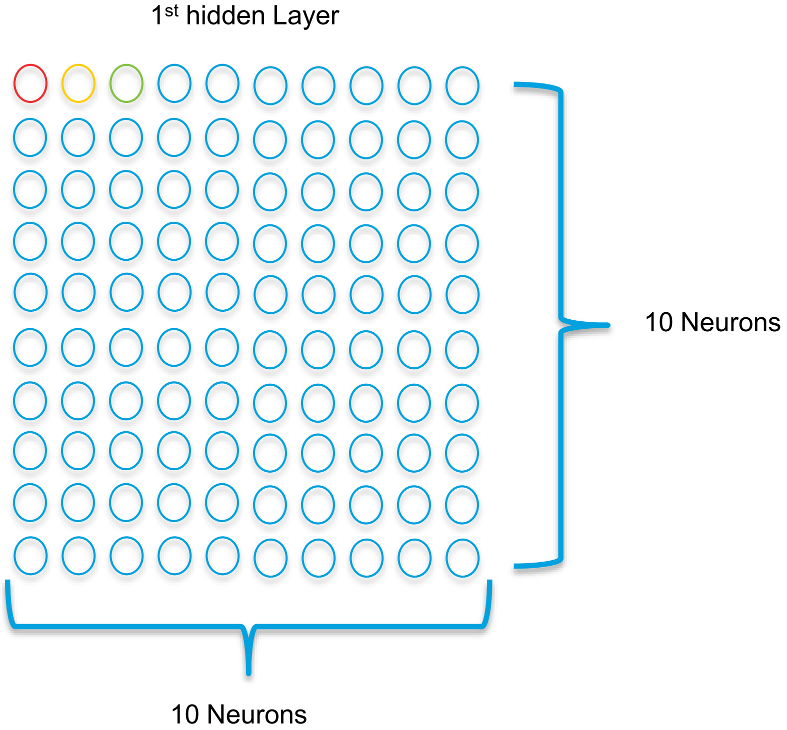
Instead, use CNN where each neuron is assigned a small local field of pixels which is same as…



1. The local neighborhood is called receptive field
2. In this example the receptive field is 3X3 pixels
3. Successive neurons get their receptive fields shifted by 1 pixel (note the overlap between red and yellow receptive fields
4. An image of size 12X12 with 3X3 local receptive fields will need 10X10 neurons in the hidden layer because

Images and First Hidden Layer in ANN





Given each neuron is associated with a local receptive region, we can show the hidden layer as two dimensional matrix of the same size as the image

Given the input image size and the local region size, the number of neurons will be 10X10