

# Computer Vision and Deep Learning

## Python Ecosystem

Bishesh Khanal, PhD

Research Scientist, NAAMII

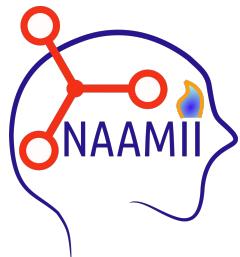
Senior AI Scientist, Fusemachines

Visiting Researcher, King College London & Imperial College London

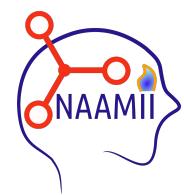
Python Users Group Nepal, Meetup #14

Kathmandu, Nepal

14 September, 2019



<https://www.naamii.com.np>



# NAAMII

<https://www.naamii.com.np>

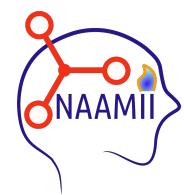
Nepal Winter School in  
AI

Research

Academic  
Social  
Outreach

**fuse** | machines  
 leapfrog

Startup  
Incubation  
&  
Industry  
Collaborati  
on



# NAAMII

<https://www.naamii.com.np>

Application deadline coming soon!  
<https://nepalschool.naamii.com.np>

Nepal Winter School in  
AI

Research

Academic  
Social  
Outreach

**fuse** | machines  
leapfrog

Startup  
Incubation  
&  
Industry  
Collaborati  
on

# Computer Vision

## Deep Learning

### Python Ecosystem for Machine Learning

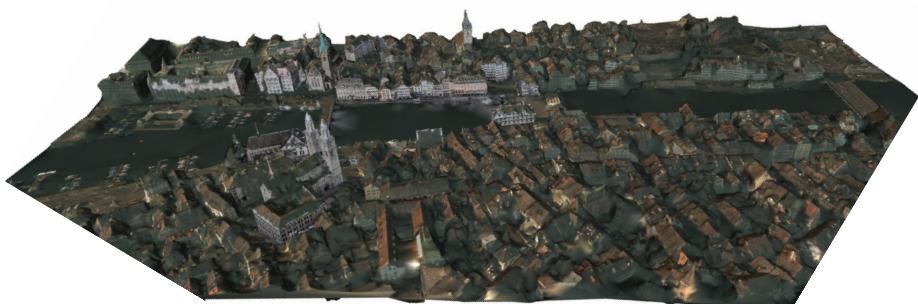
# Computer Vision: 3D Modeling

- Automatically compute 3D model from 2D images



# Computer Vision: 3D Modeling

- ❑ Automatically compute 3D model from images



3D Model of a part of Zurich modeled from images – source VarCity, CVL –ETH Zurich

# Computer Vision: Traffic Understanding

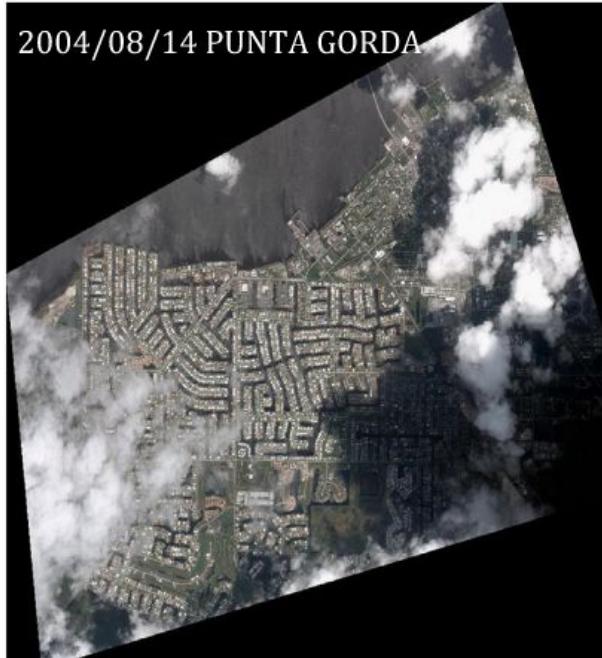


# Computer Vision: Disaster Area/Damage Identification

a



b



c



**Fig. 1.** (a) Pre-storm image (source: DigitalGlobe Co., Ltd.). (b) Post-storm Image (source: DigitalGlobe Co., Ltd.). (c). Field investigation information (source: [Womble, 2005](#))

# Computer Vision: Disaster Area/Damage Identification



**Fig. 2.** (a) A sample building before cyclone damage. (b) After cyclone damage. (c) Non damaged portion marked in red outline. (d) Ground truth data of the same building (source: [Womble, 2005](#)). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Article

# Detection of Urban Damage Using Remote Sensing and Machine Learning Algorithms: Revisiting the 2010 Haiti Earthquake

Austin J. Cooner \*, Yang Shao and James B. Campbell

Virginia Tech Department of Geography, 115 Major Williams Hall 220 Stanger St., Blacksburg, VA 24060, USA;  
yshao@vt.edu (Y.S.); jayhawk@vt.edu (J.B.C.)

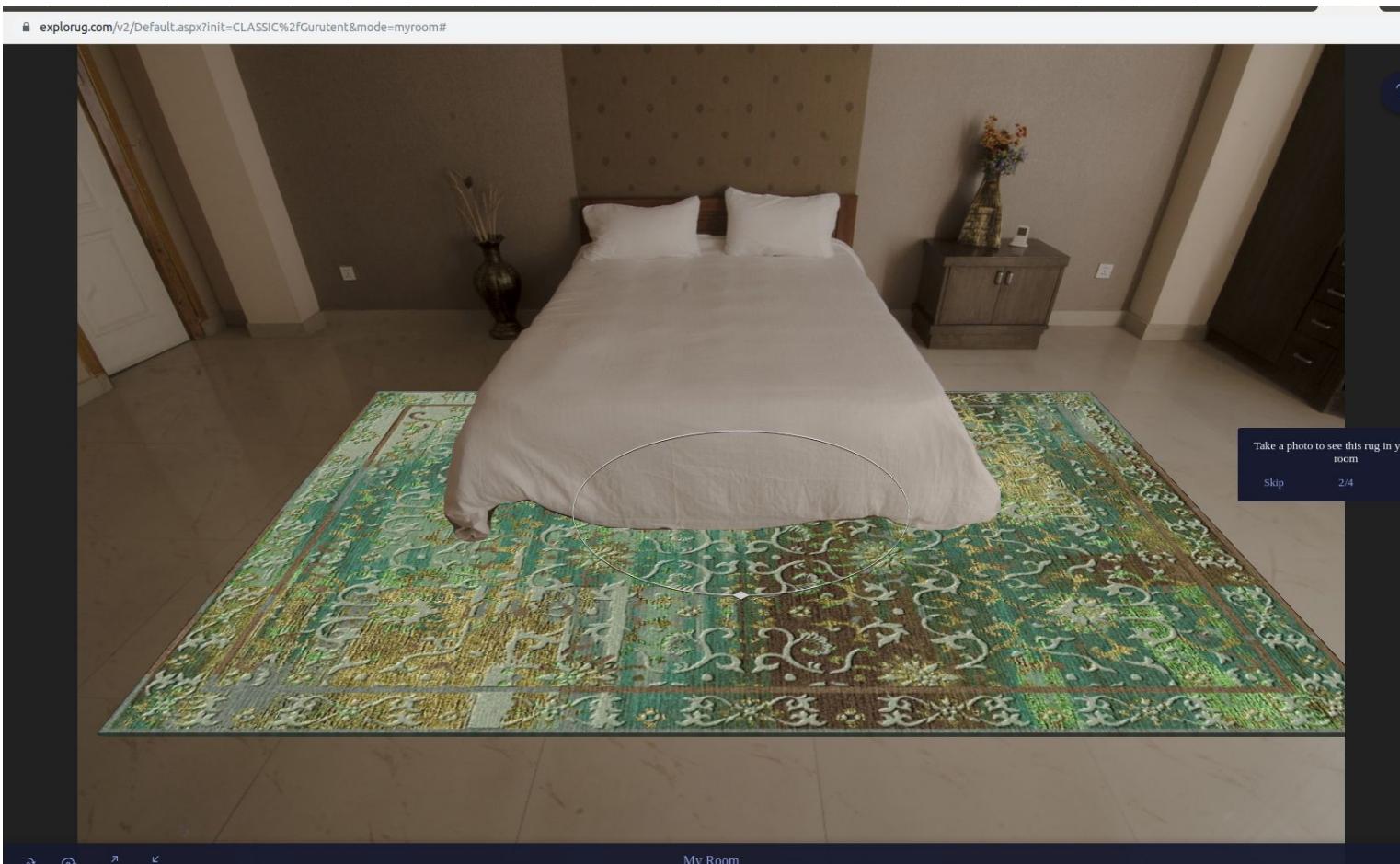
\* Correspondence: cooner@vt.edu; Tel.: +1-210-870-7909

Academic Editors: Roberto Tomas, Zhenhong Li and Prasad S. Thenkabail

Received: 5 August 2016; Accepted: 17 October 2016; Published: 20 October 2016

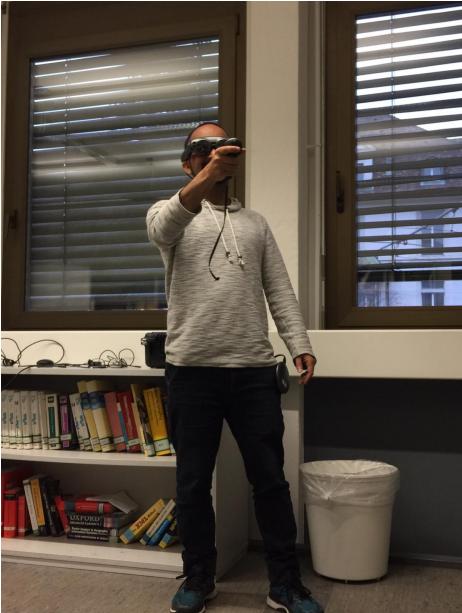
**Abstract:** Remote sensing continues to be an invaluable tool in earthquake damage assessments and emergency response. This study evaluates the effectiveness of multilayer feedforward neural networks, radial basis neural networks, and Random Forests in detecting earthquake damage caused by the 2010 Port-au-Prince, Haiti 7.0 moment magnitude ( $M_w$ ) event. Additionally, textural and structural features including entropy, dissimilarity, Laplacian of Gaussian, and rectangular fit

# Computer Vision: Augmented Reality

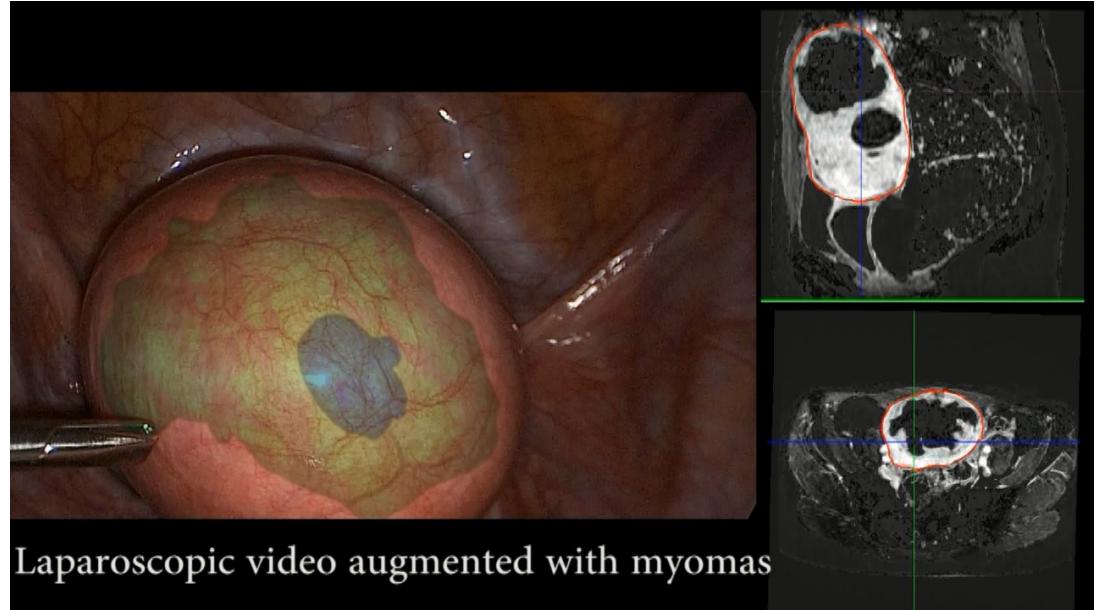


# Computer Vision: Virtual/Augmented/Mixed Reality

## Mixed Reality and Virtual Reality

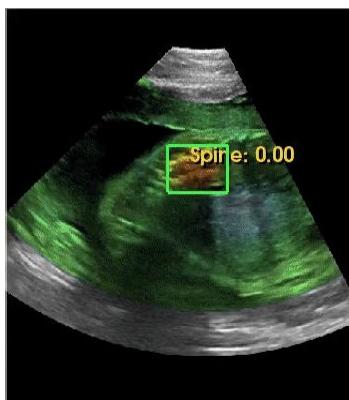
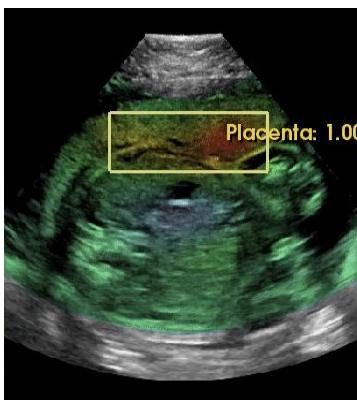
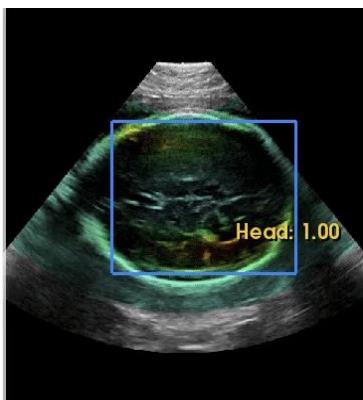
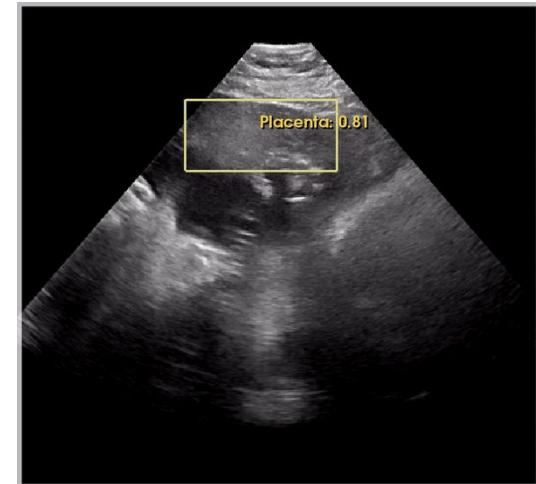
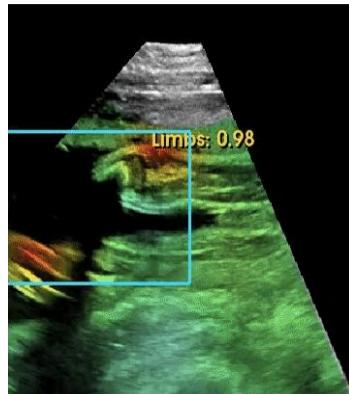
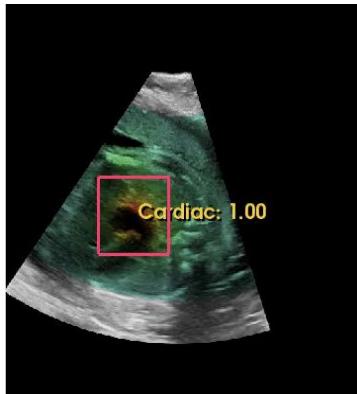


MagicLeap demo



ENCoV - France

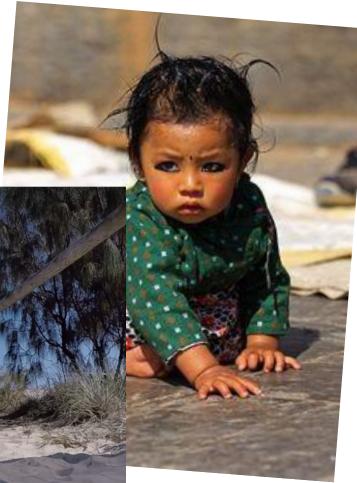
# Computer Vision: Medical Imaging



# How do we see ?



# How do we LEARN to see ?



# How do we learn to see ?

We move in order to see and see in order to move  
William Gibson



source: Traveltourtrek  
Nepal



We live in a 3D world

# How do we learn to see ?

We move in order to see and see in order to move

William Gibson



source: Traveltourtrek  
Nepal



We live in a 3D world

Three pillars of AI - data, learning and knowledge

-Fei Fei Li

# How do we learn to see ?

We move in order to see and see in order to move

William Gibson



source: Traveltourtrek  
Nepal

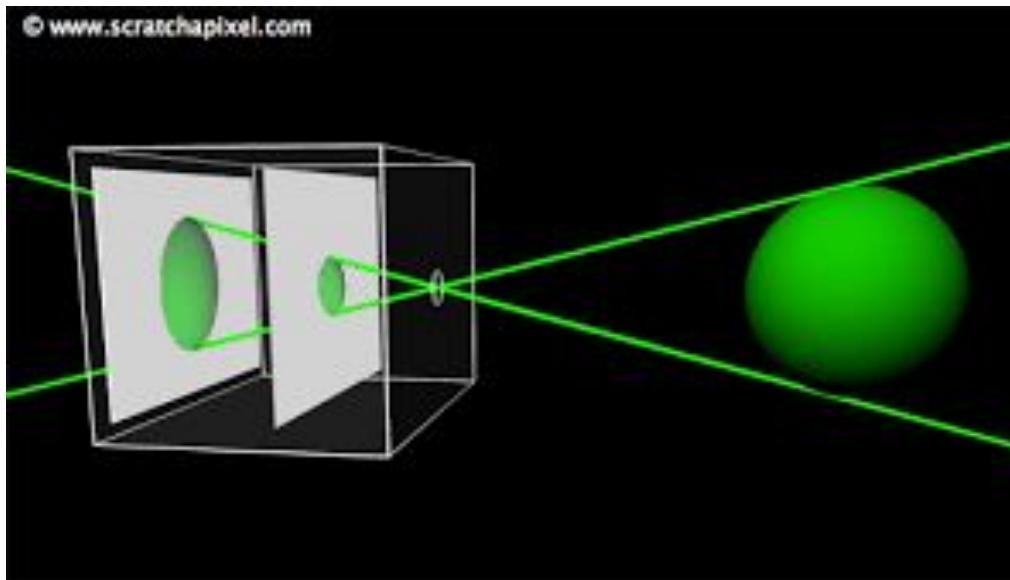
Three pillars of AI - data, learning and knowledge  
-Fei Fei Li



We live in a 3D world

Not just about the eye, a mere sensor.  
Human Vision is done in the Brain!

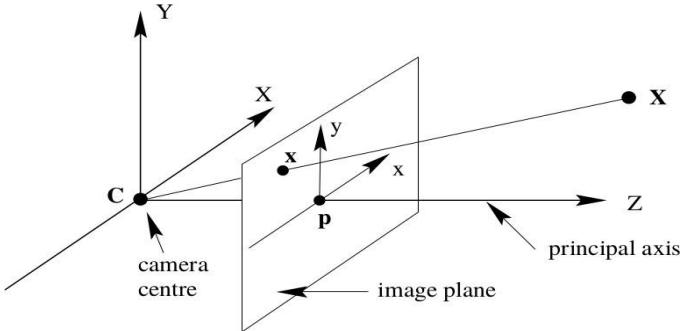
# From 3D world to 2D images



# Image Formation



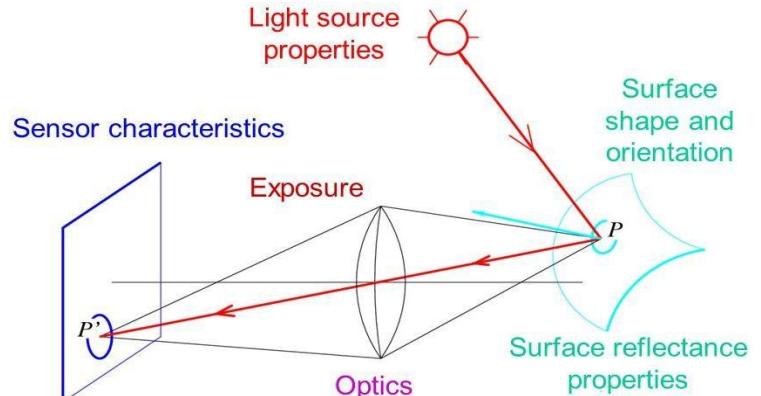
What we see



Source: Hartley, Richard, and Andrew Zisserman. Multiple view geometry in computer vision. Cambridge university press, 2003.

192	191	191	191	191	191	191	193	193	191	190	192	194	193	192	199	185	184	179	179	177	
191	191	192	194	194	194	193	193	193	192	193	194	196	197	196	193	191	190	187	181	189	177
192	192	192	192	193	193	194	195	194	196	195	194	195	195	194	199	188	186	179	174	176	
188	189	191	193	194	196	195	196	195	195	195	194	196	196	192	188	186	179	174	175		
185	187	191	194	195	195	194	194	195	196	195	195	195	195	192	188	184	180	175	175		
184	186	188	193	193	193	193	194	194	194	194	194	194	194	194	197	197	197	197	197		
178	182	188	193	193	192	191	191	192	188	187	187	185	184	183	181	179	179	174	173		
173	175	183	188	186	184	185	185	186	186	184	182	181	180	179	178	178	175	175	176		
167	170	176	180	181	180	180	180	179	179	178	178	177	177	176	177	174	174	175	177		
161	164	167	170	174	175	174	173	173	174	174	174	174	174	172	172	174	174	175	177		
154	157	159	162	167	170	166	167	168	169	171	171	171	170	170	170	172	172	176	176		
149	150	152	153	154	155	155	156	156	156	157	157	157	157	158	160	160	160	160	160		
138	137	141	145	147	151	154	153	153	154	156	159	163	166	167	167	170	173	175	176		
139	135	135	137	138	140	142	144	144	151	154	156	157	160	164	166	167	170	172	175	177	
142	137	134	138	139	139	141	146	152	156	157	157	159	164	167	168	170	172	174	176		
141	137	136	140	141	143	144	148	156	157	157	161	165	167	168	169	171	173	175			
142	139	137	137	138	139	142	144	159	157	157	159	162	165	167	170	172	174	175			
144	141	143	142	140	139	142	144	148	156	157	159	162	163	167	168	171	172	173			
144	143	142	139	137	138	142	144	148	156	159	161	164	163	163	167	169	170	171			
143	142	141	137	137	139	144	147	148	155	158	161	163	163	165	167	168	169	170	171		
144	142	141	140	139	139	141	146	159	155	160	161	162	163	167	168	168	169	170	171		
147	144	143	142	141	142	145	147	154	158	160	162	162	164	168	169	170	170	171			
147	145	145	145	144	143	143	145	150	155	158	160	163	167	169	169	170	172	172	171		
150	149	148	148	147	146	144	144	148	152	152	153	153	153	153	153	153	153	153	153		
156	150	147	146	147	149	149	151	155	159	165	162	164	164	169	170	172	172	171	172		
161	156	153	151	150	150	153	154	154	156	168	166	164	163	167	170	171	172	171	171		
164	161	157	154	153	152	151	154	157	162	164	164	163	164	164	165	171	172	173	172		
168	165	161	157	156	154	152	154	159	161	159	161	162	165	162	163	168	171	172	173		

What machines see



Source: L. Fei Fei

# What do we see ?



# What do we see ?



Brain known to process at different levels.

Marr's 3 levels:

- Primal sketch : edges, corners, regions : “features”
- 2.5 D sketch : texture, depth concept
- 3D model : the scene as we see it in a 3D world

# Hand Engineering to Extract Features

Traditional approach (very popular: 1990's - 2010)

- Extract features designed by engineers/researchers
- Use them to learn higher level semantics

# Hand-Engineered Feature Extraction



# Hand-Engineered Feature Extraction: Edge Detection



# Hand-Engineered Feature Extraction: Threshold



# Hand-Engineered Feature Extraction: Harris Corner



# Hand-Engineered Feature Extraction: Harris Corner



Many more!  
SIFT  
SURF  
ORB  
LBP  
Convolutional Filters  
...

Easy to understand  
No big data need  
Fast if designed well

# It is not an easy task!

---

Viewpoint variation



Scale variation



Deformation



Occlusion



Illumination conditions



Background clutter



Intra-class variation



---

Source: <http://cs231n.github.io/classification>

Features depend on imaging conditions and hard to design “invariant” features  
Relatively strong domain knowledge required

Computer Vision

Deep Learning

Python Ecosystem for Machine Learning

# Learn from the data needed features & representation

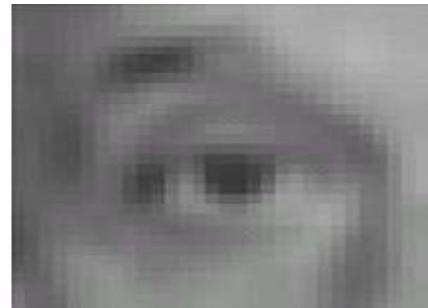
Building block of feature extraction: convolution/filter/kernel



Original

$$\frac{1}{9} \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

Filter

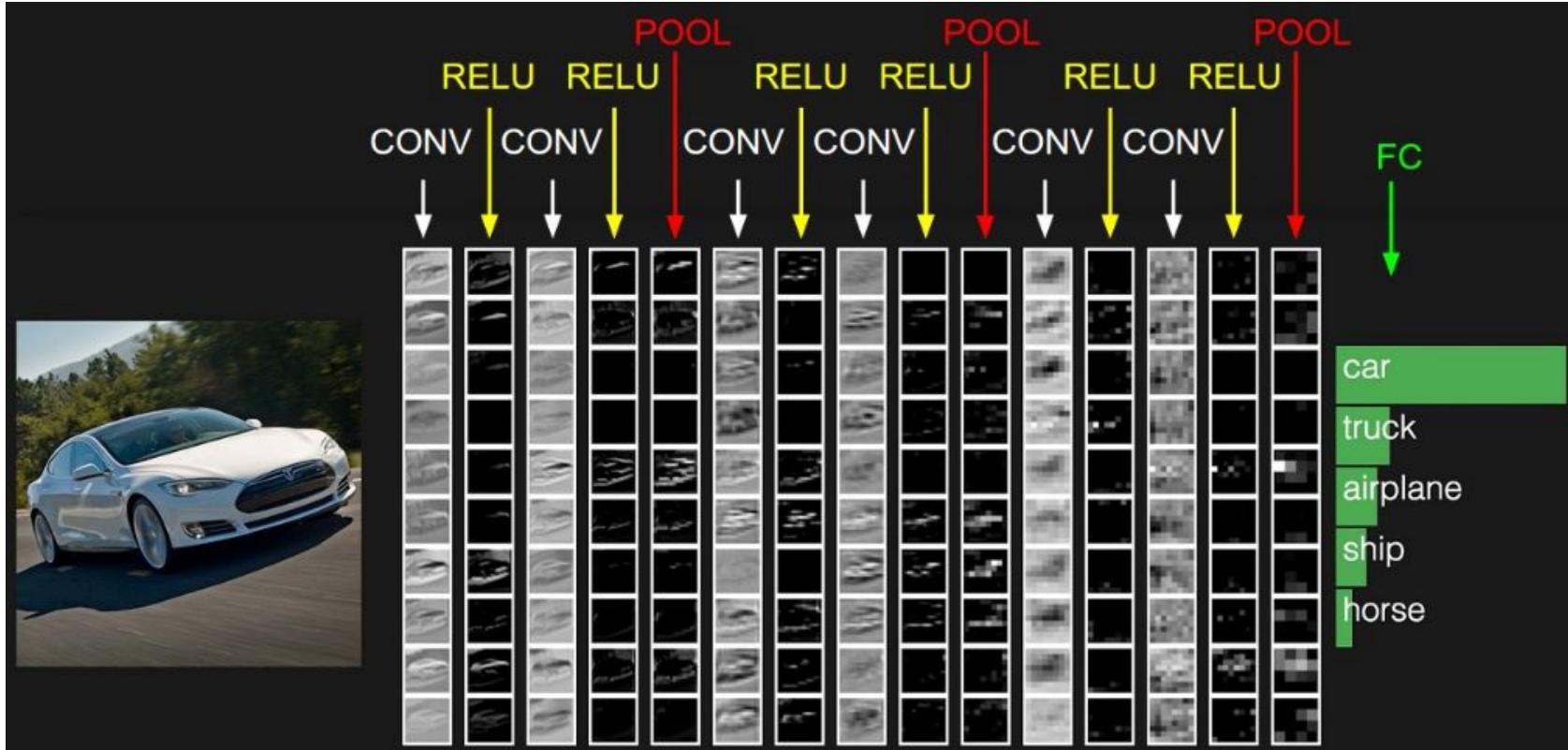


Result

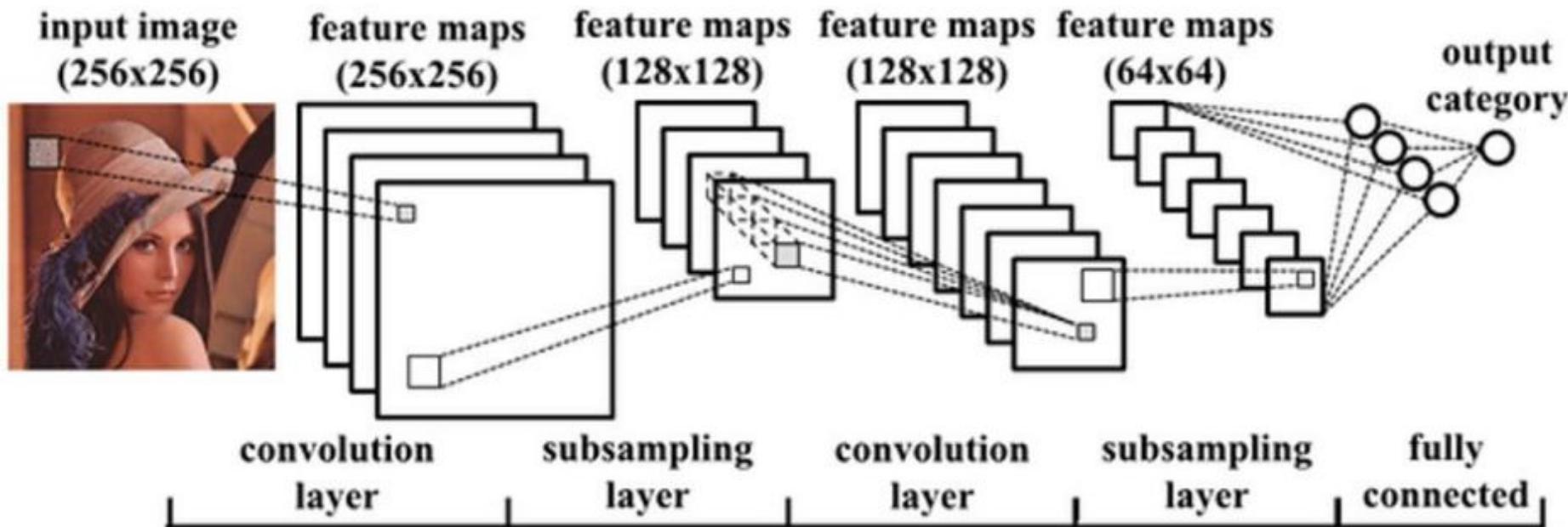
(blur with a box filter)

Learn these filters/weights from lot of data on a given task as required

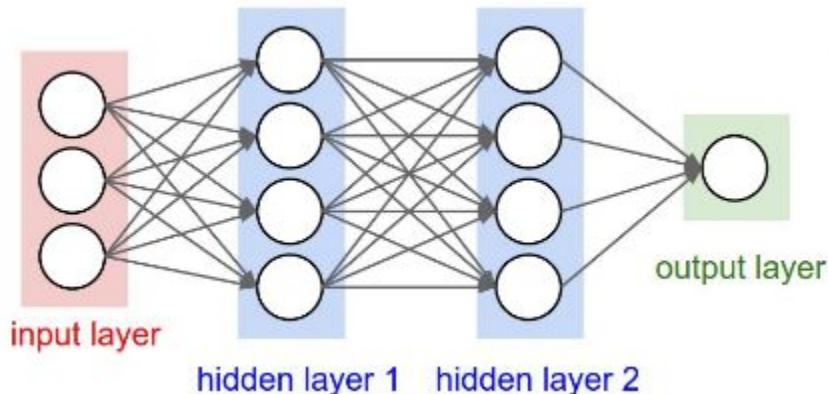
# Deep Convolutional Neural Networks most popular



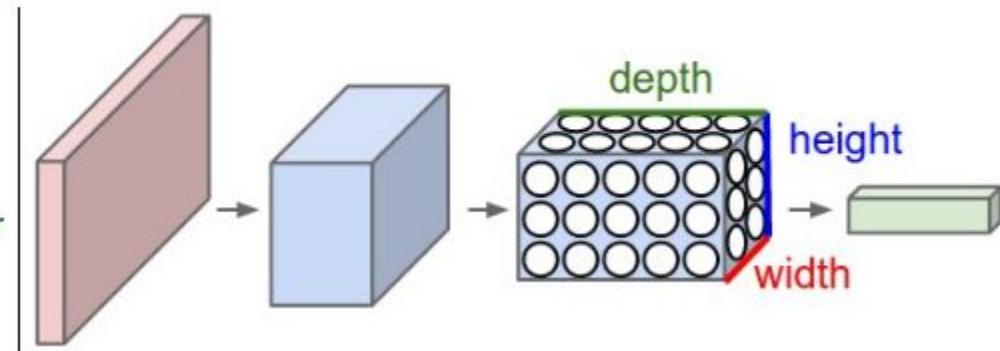
# Deep Convolutional Neural Networks most popular



# CNNs



Regular 3 layer network



Convnet arrangement in 3D

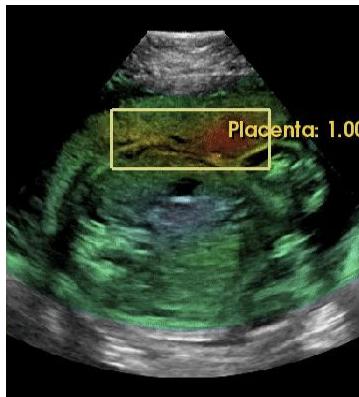
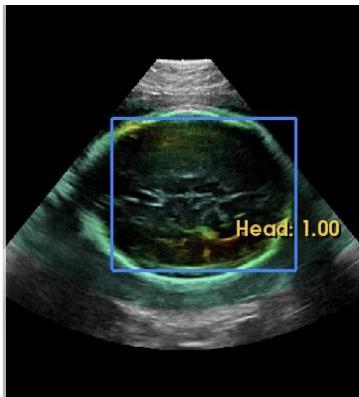
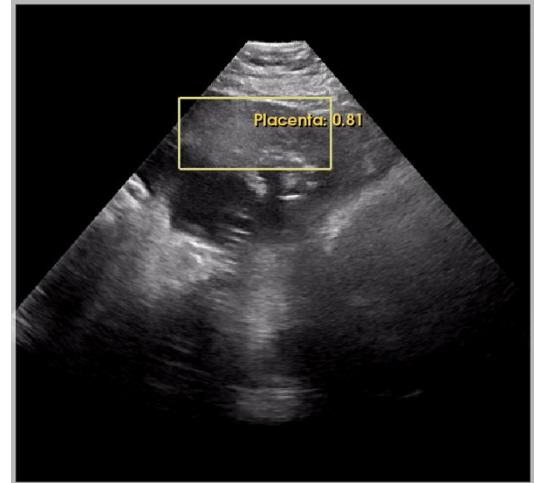
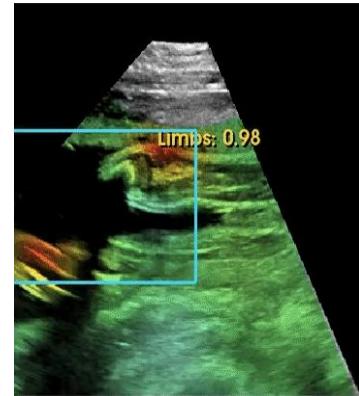
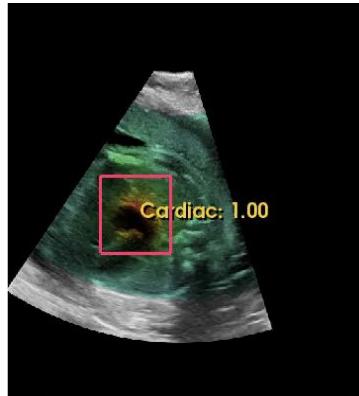
Transforms input 3D volume into output 3D volume

E.g. Input of 3 Channels R,G,B; image of size height X width

# Example Applications



# Example Applications



# Are neural networks and CNNs new ?

NNs from 1950s, CNNs from 70s, but optimizing them was a challenge!!

How come they are so popular now ?

# Why Deep Learning worked now?

10 years rapidly expanding use of Internet: BIG DATA availability

Massive improvement in Hardware (GPUs)

Advancement in optimization techniques

Great open-source software frameworks to run on those GPUS (mostly python based)

# Why Deep Learning worked now?

10 years rapidly expanding use of Internet: BIG DATA availability

Massive improvement in Hardware (GPUs)

Advancement in optimization techniques

Great open-source software frameworks to run on those GPUS (mostly python based)

**Not everything solved!!**

Less data and learn, Unsupervised learning, best representation

Generalization

Interpretability

Computer Vision

Deep Learning

**Python Ecosystem for Machine Learning**

# Python for Scientific Computing & Data Science

```
import cv2
import numpy as np

img = cv2.imread('image1.jpg')
gray= cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

gray = np.float32(gray)
dst = cv2.cornerHarris(gray,2,3,0.04)

#result is dilated for marking the corners, not important
dst = cv2.dilate(dst,None)

# Threshold for an optimal value,
# it may vary depending on the image.
#img[dst>0.01*dst.max()]=[0,0,255]

img[dst>0.001*dst.max()]=[0,0,255]

#cv2.imshow('dst',img)
#if cv2.waitKey(0) & 0xff == 27:
#    cv2.destroyAllWindows()

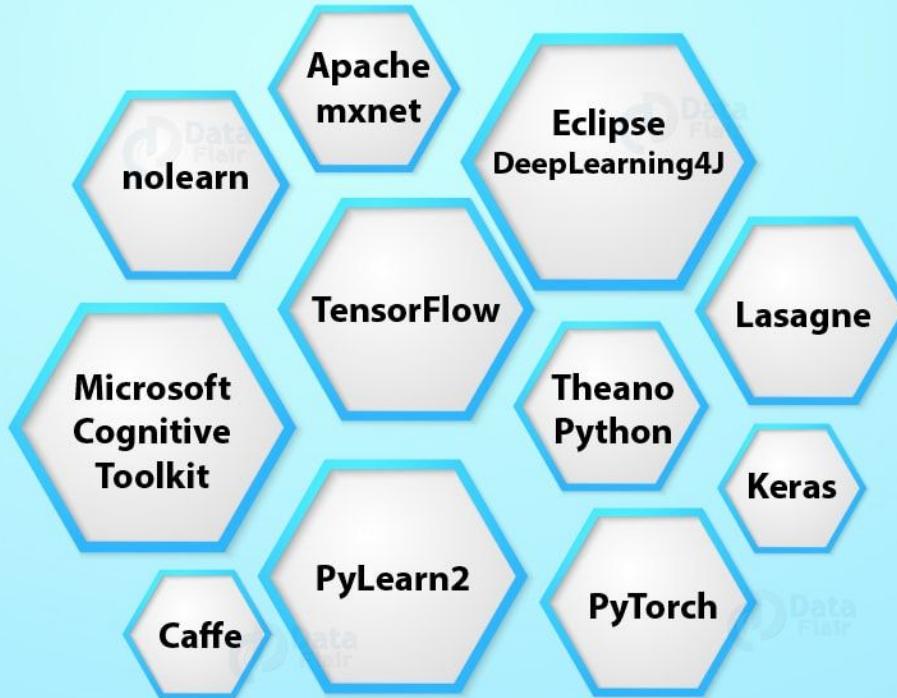
cv2.imwrite('harris-corners-image1.jpg',img)
```



```
from mmdet.apis import init_detector, inference_detector, show_result
import mmcv
config_file = 'mmdetection/configs/faster_rcnn_r50_fpn_1x.py'
checkpoint_file = '../mmdetection/checkpoints/faster_rcnn_r50_fpn_1x_20181010-3d1b3351.pth'
model = init_detector(config_file, checkpoint_file, device='cuda:0')
img = 'results/Kathmandu-Traffic.jpg'
result = inference_detector(model, img)
show_result(img, result, model.CLASSES, out_file='results/Kathmandu-Traffic-out.jpg')
```



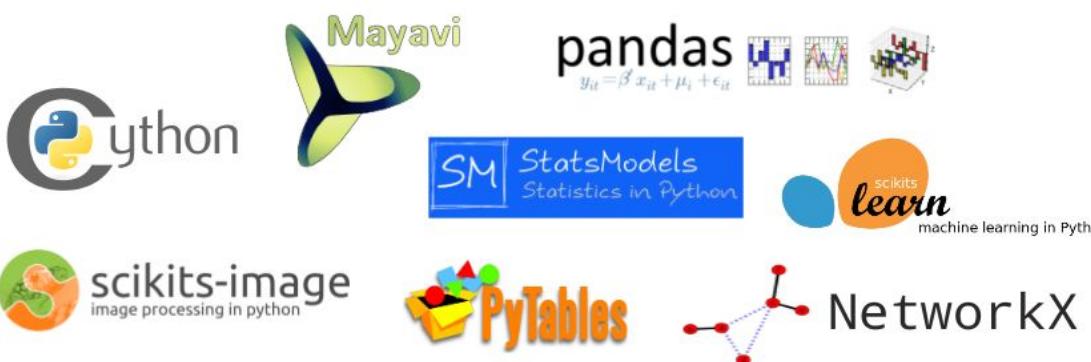
# Python for Deep Learning



11

## Deep Learning With Python Libraries and Frameworks

# Python for Scientific Computing & Data Science



# Second Nepal Winter School in AI

10-20 December, 2019

Pokhara, Nepal

NEUROSCIENCE

PROBABILITY  
& STATISTICS

GRAPHICAL MODELING

AI &  
SOCIETY

DEEP  
LEARNING

COMPUTER VISION

Application for Participation now open !!!

Limited Scholarships and Travel Grants Available !!!

Internships and vacancies at NAAMII announcing soon

<https://www.naamii.com.np/>

[https://twitter.com/naamii\\_nepal](https://twitter.com/naamii_nepal)

<https://www.facebook.com/naamiiNepal>