Lecture 1: Introduction to Deep Learning

Source: From Pinwest

Learning ：

The acquisition of knowledge or skills through experience, study, or by being taught.

Source: from google dictiyynary

"Acquires the awareness and perception of the facts and adapts to the ever changing environment to make sound judgment and utilization of it... and to make machine to do that" … HL

Deep Learning ：

The technique based on convolutional neural networks which achieves the acquisition of knowledge or skills through experience (training), study (comparison, investigation and analysis), or by being taught (training the neural networks).

Two Keys:

1. Neural Networks

2. Convolution (convolutional neural networks)

Software Tools for Deep Learning

Software tools:

1. OpenCV http://opencv.org/

Free for both academic and commercial use, with C++, C, Python and Java interfaces, originally developed at Intel designed for multi-core processing. Enabled with OpenCL, has more than 47K user community and exceeding 9 million downloads, latest version 3.2.

2. Tensorflow

Developed by google brain team. TensorFlow is an open-source software library for machine learning released in 2015. It is a system for building and training neural networks to detect and decipher patterns and correlations.

Introduction to Tensorflow

Developed by google brain team. TensorFlow is an open-source software library for machine learning released in 2015. It is a system for building and training neural networks to detect and decipher patterns and correlations.

Tutorial on line

http://www.oreilly.com/ai/deep-learning-tensorflow.html?utm_source=google&utm_medium=cpc&utm_campaign=TensorFlow&utm_term=tensorflow%20tutorial&utm_content=220150902641&gclid=CjwKCAjw9O3NBRB3EiwAK6wPTywgkgg6uD45RHHE77IMHw8KV2MDr275KicGXTQtQqLrnR61LrswSRoCAqsQAvD_BwE

TensorFlow 1.3 has arrived!

API for C/C++, Python, Java etc.

https://github.com/tensorflow

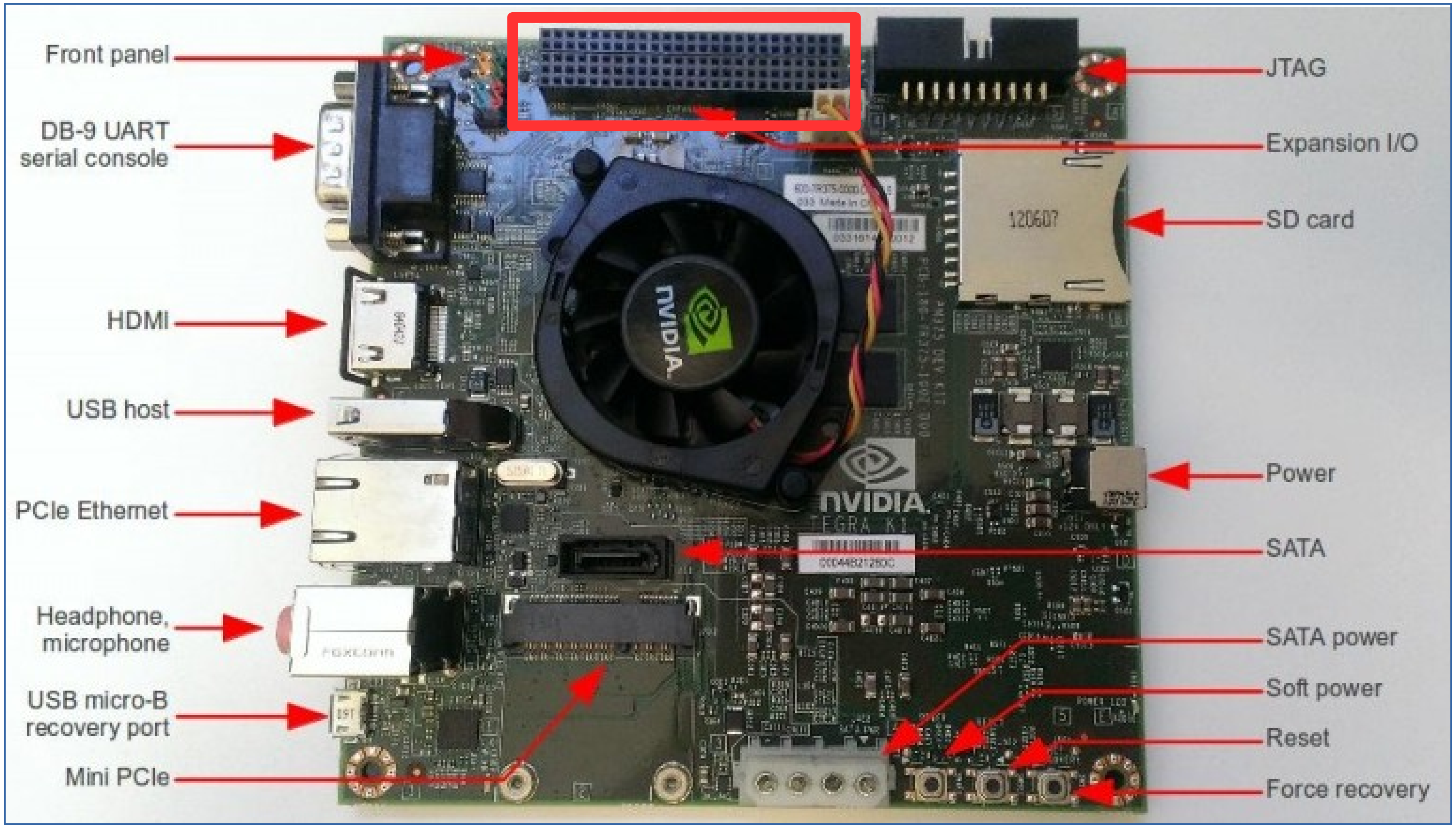
Youtube example:
https://www.youtube.com/watch?v=uHaKOFPpphU

Embedded Platform for Product Deployment

Jetson tk1 GPU Embedded Platform: cross over like platform, either as an embedded platform or desktop machine

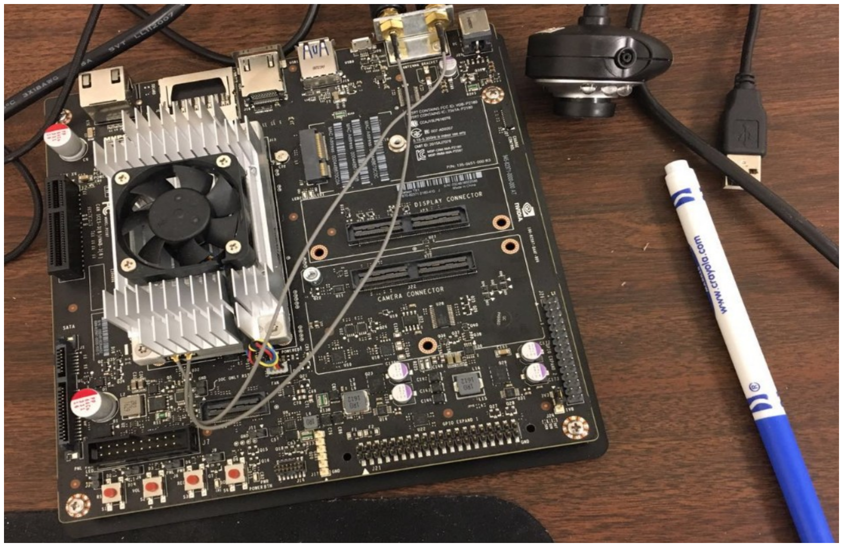
1. http://elinux.org/Jetson_TK1

Training not on this platform



Use putty on my ubuntu laptop to run Jetson

Embedded Platform TX1SVR

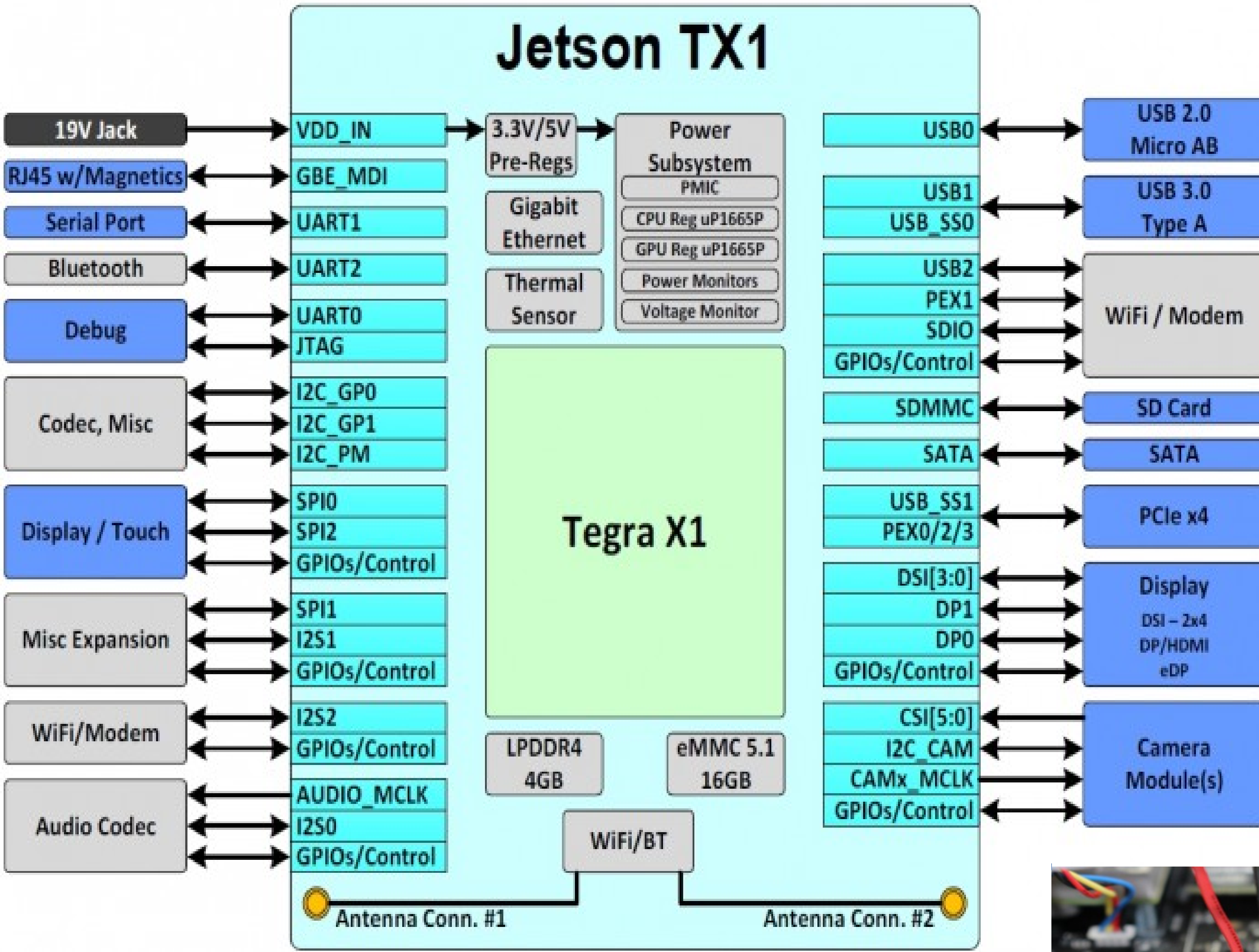


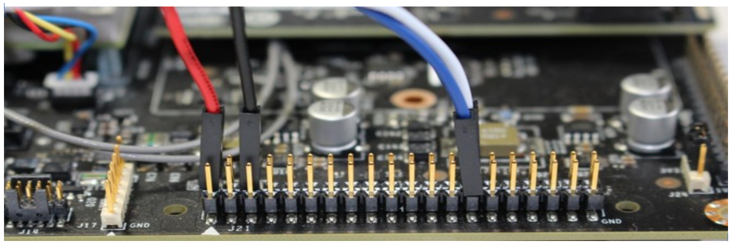
https://developer.nvidia.com/embedded/buy/jetson-tx2

http://elinux.org/Jetson/TX1_SPI

NVIDIA Maxwell ™, 256 CUDA cores
Quad ARM® A57/2 MB L2
video 4K x 2K 30 Hz Encode (HEVC) 4K x 2K 60 Hz Decode (10-Bit Support)
4 GB 64 bit LPDDR4 25.6 GB/s
2x DSI, 1x eDP 1.4 / DP 1.2 / HDMI
Up to 6 Cameras (2 Lane) CSI2 D-PHY 1.1 (1.5 Gbps/Lane)
Gen 2 | 1x4 + 1x1
16 GB eMMC, SDIO, SATA
UART, SPI, I2C, I2S, GPIOs

TX1SVR Architecture Overview





About the Instructor (1)

Harry Li

Professor

(408) 924-4060

hua.li@sjsu.edu

ENG 267A

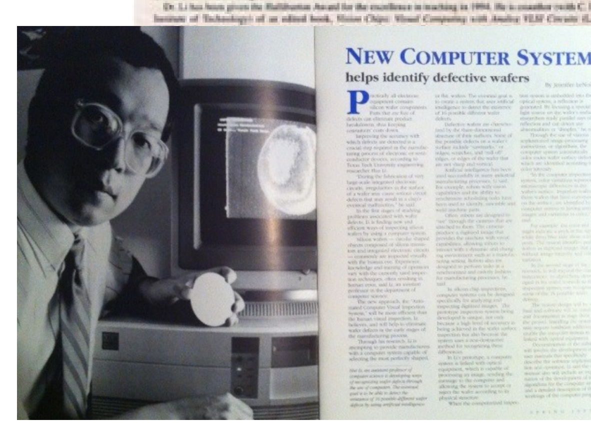
Professor of Computer Engineering, with research expertise in Artificial Intelligence, Computer Vision and Embedded Systems who started his higher education career as Assistant Professor in Computer Science Department, College of Engineering at Texas Tech University in 1989, and then Associate Professor in 1994, and joined the San Jose State University in January 1997.

IEEE TRANSACTIONS ON
COMPONENTS, PACKAGING, AND MANUFACTURING TECHNOLOGY
PART C: MANUFACTURING

JANUARY 1996 VOLUME 19

Foreword:
Neural Networks and Fuzzy Logic

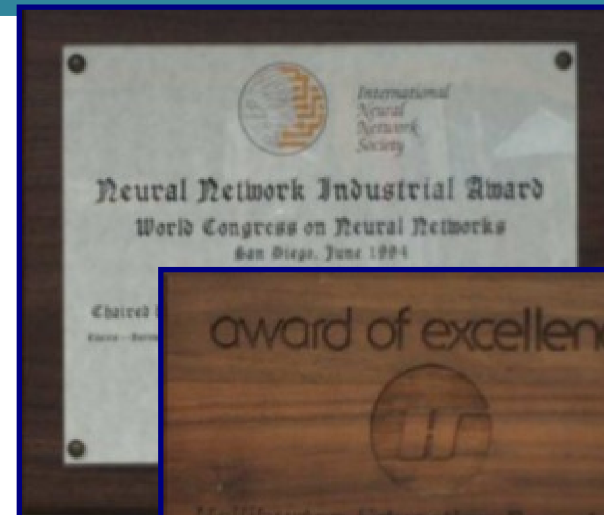
Guest Editor for the emerging technology section of IEEE Transactions on CPMT (Components, Packaging, and Manufacturing Technology), one of the top ranking Professional Technical Journals in the field. He was invited contributor to the Encyclopedia of Electrical and Electronic Engineers (On Neural Fuzzy Control techniques for semiconductor equipment control applications, pp. 154- 157, Vol. 8, Fr-Hi, ISBN 0-471-13946-7, John Wiley and Sons Inc.)

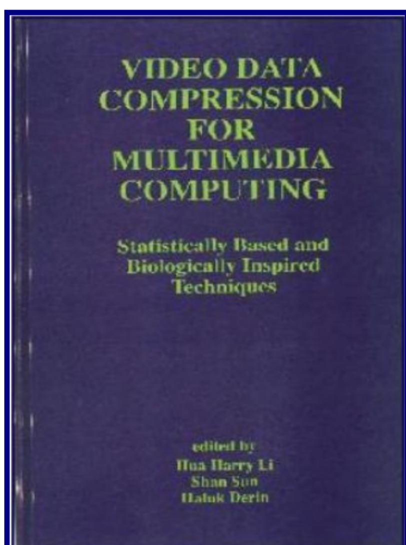


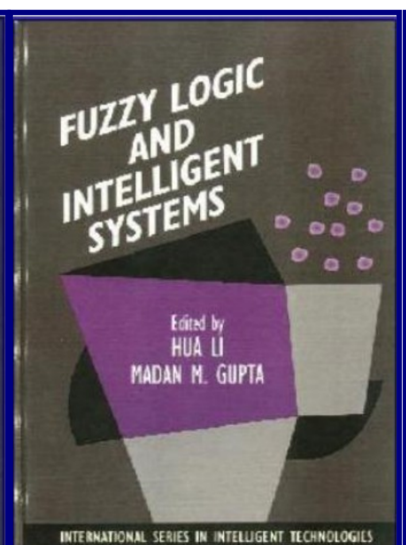
NEW COMPUTER SYSTEM
helps identify defective wafers

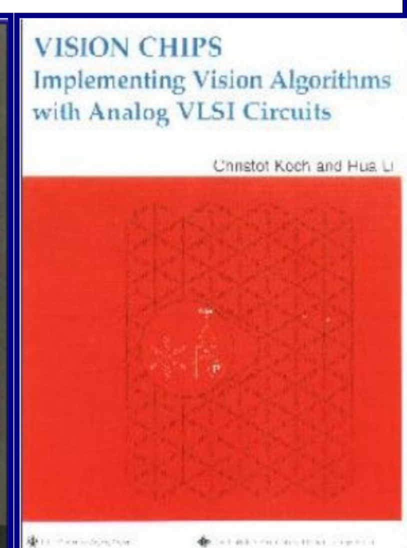
About the Instructor (2)

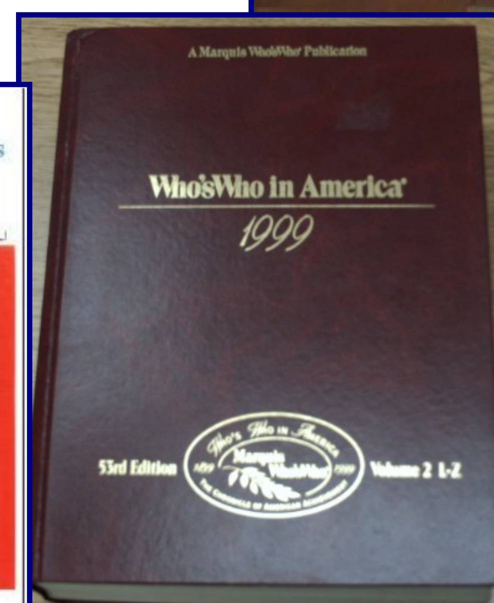
Dr. Li has active and extensive High Tech industry experience in the Silicon Valley including working as consulting principle engineer, senior system architect, principle technologist in a range of companies from start-up to a major telecom public traded company. Dr. Li has been actively exploring and pursuing the joint development of technology innovation and entrepreneurship throughout his career. He has been actively teaching in the Silicon Valley companies including Lockheed, KLA-Tencor, Ebay etc.











About the Instructor (3)

Tony Xu M.B.S., ECE, M.S., ECE, Senior Engineering Training Expert, Senior Engineer at Intel

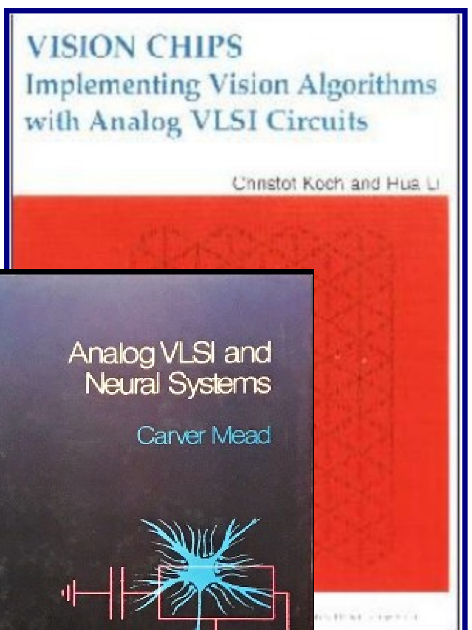
Over 20 years experience in software engineering. His professional expertise includes system performance improvement, great user experience, micro to system level performance analysis. In the last five years, he has focused his effort on open source projects, including OpenCV, 3D camera, FPGA machine learning acceleration and OpenCL. Tony has extensive knowledge of system architecture – FSB, CSI, cache architecture, coherence protocol, APIC controller, I/O controller, memory controller. As a hands-on software engineer, he loves to code. He is very fluent in C/C++, Python etc, and Tony graduated from SJSU with a master’s degree in Computer Engineering.

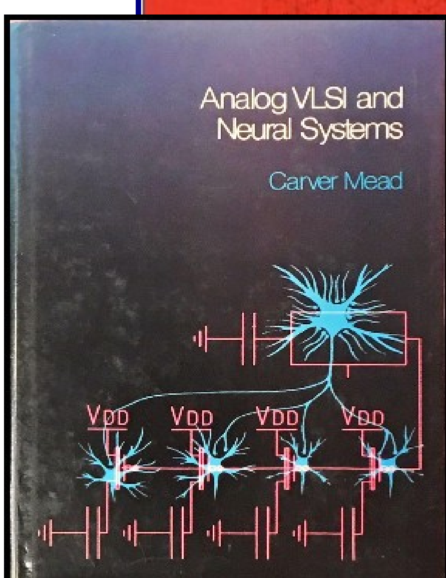
Joshua Zhang B.A., Corporate Executives. born in Lubbock, TX but spent most of his life inDetroit, MI before attending Harvard University where he studied Economics and graduated with Honors. Josh began his career at Goldman Sachs in the Real Estate Finance Group where he closed approximately $9 billion of commercial real estate debt transactions spanning traditional mortgage-backed loans, bank loans, and mezzanine financings. Currently, Josh is the Vice President of Acquisitions at Four Corners Property Trust (NYSE:FCPT), a $1.8bn net-lease REIT where he is responsible for sourcing acquisition pipeline, executing deals, and managing the company's portfolio. Josh is the member of the Advisory Board to CTI One Training Program and Instructor on entrepreneurship program.

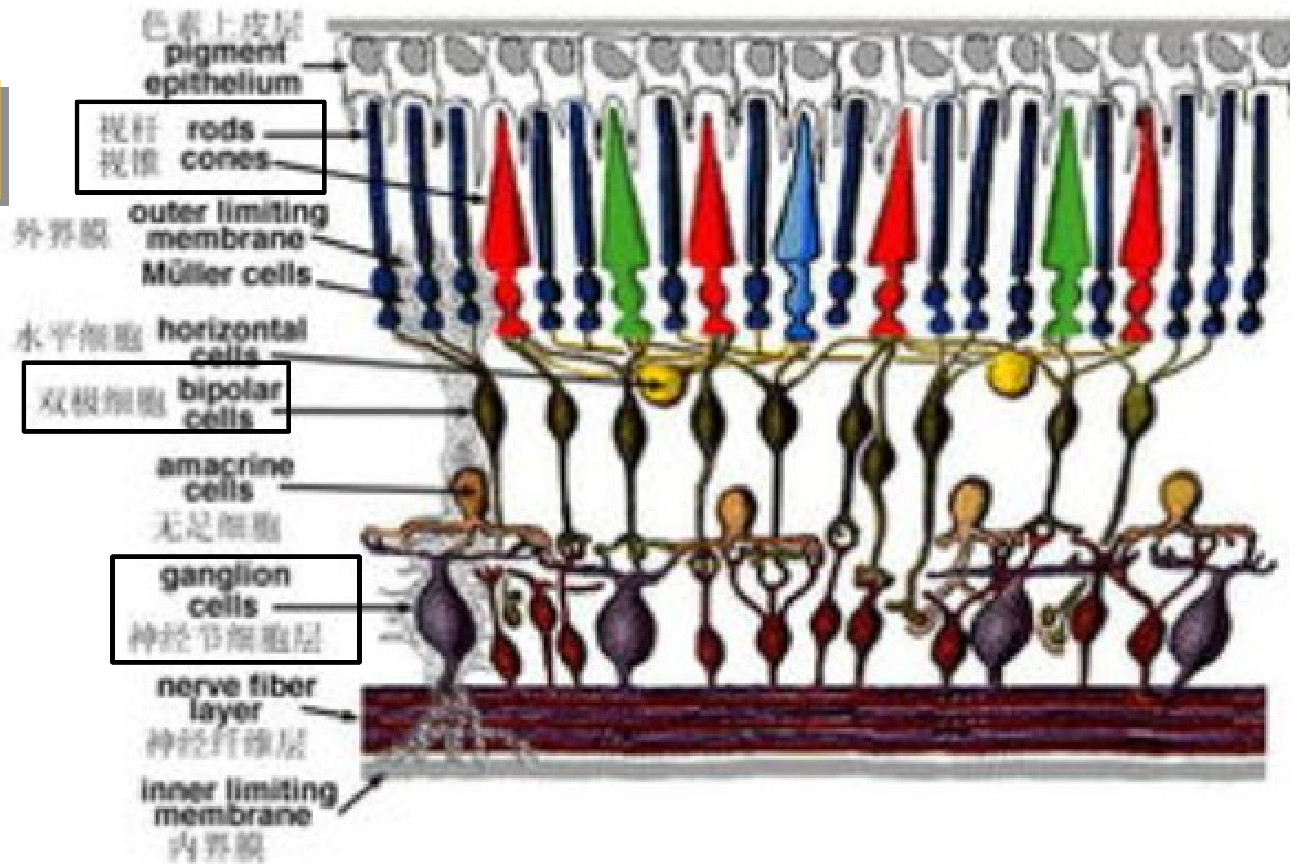
Biologically Inspired Techniques

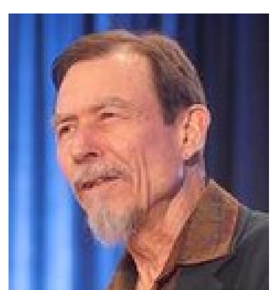
Rod, cone, bipolar cells and ganglion cells

Joint edited book with Professor Koch and myself









Prof. Mead

Prof. Koch

Prof. H. Li

VLSI Implementation

Vision Important Deep Learning Goal

Total human brain cells: “This was accomplished by Azevedo et al. (2009), who found that the adult male human brain, at an average of 1.5 kg, has 86 billion neurons and 85 billion non-neuronal cells” from google

~= 100 Billion = 100 x 10 ^ (9) = 10 ^ (11) for neural cells

More than 60% human brain cells are for vision related processing

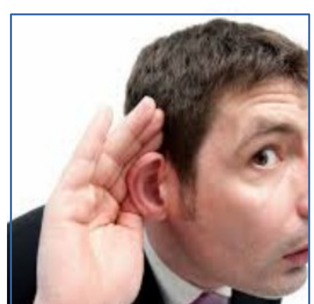
See

=>

Touch

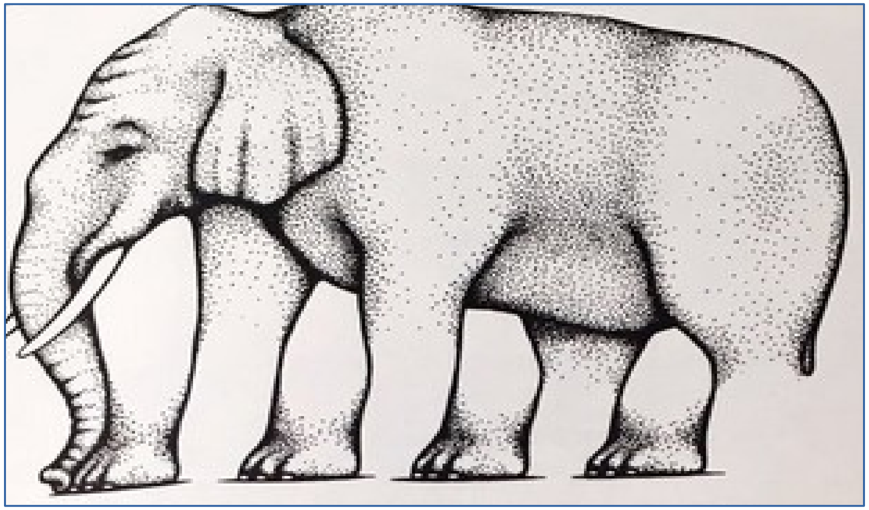
Taste

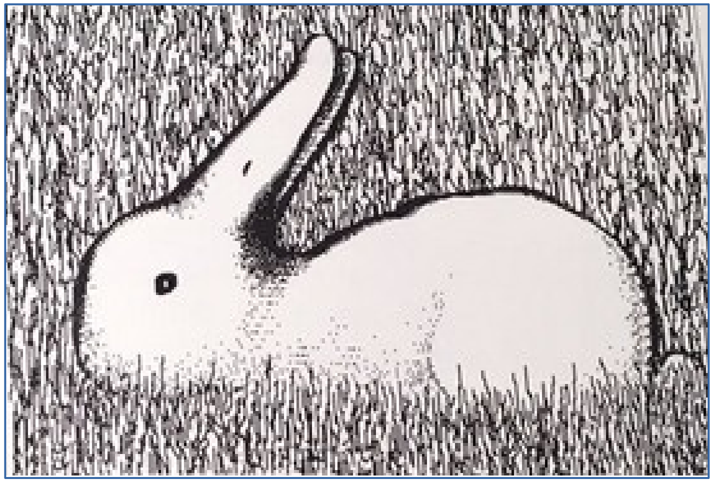
Smell



Hear

Vision Perception Challenges





Human Vision System

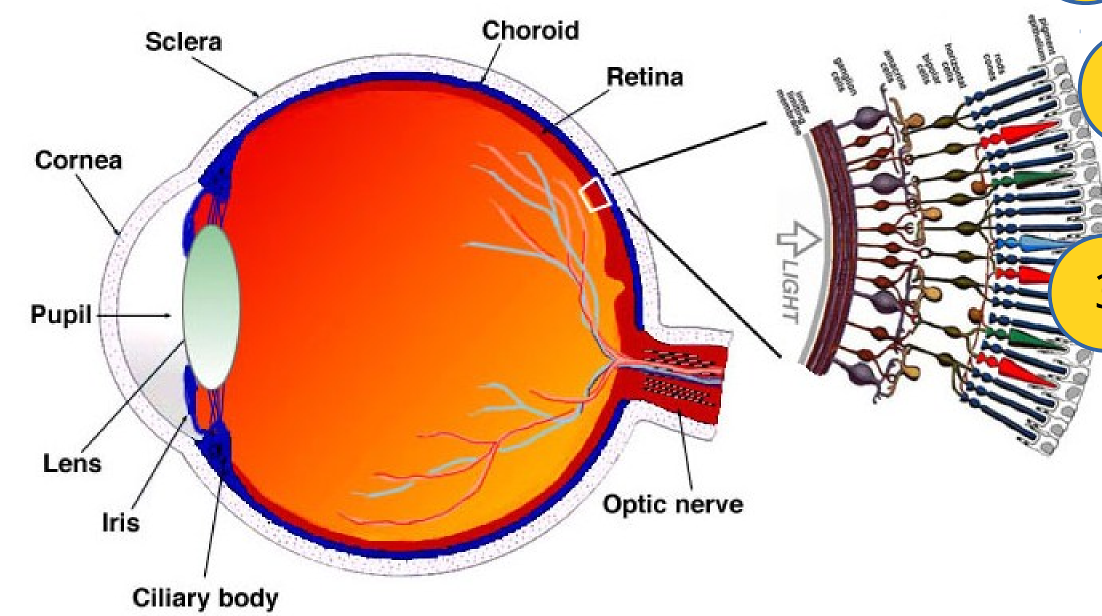
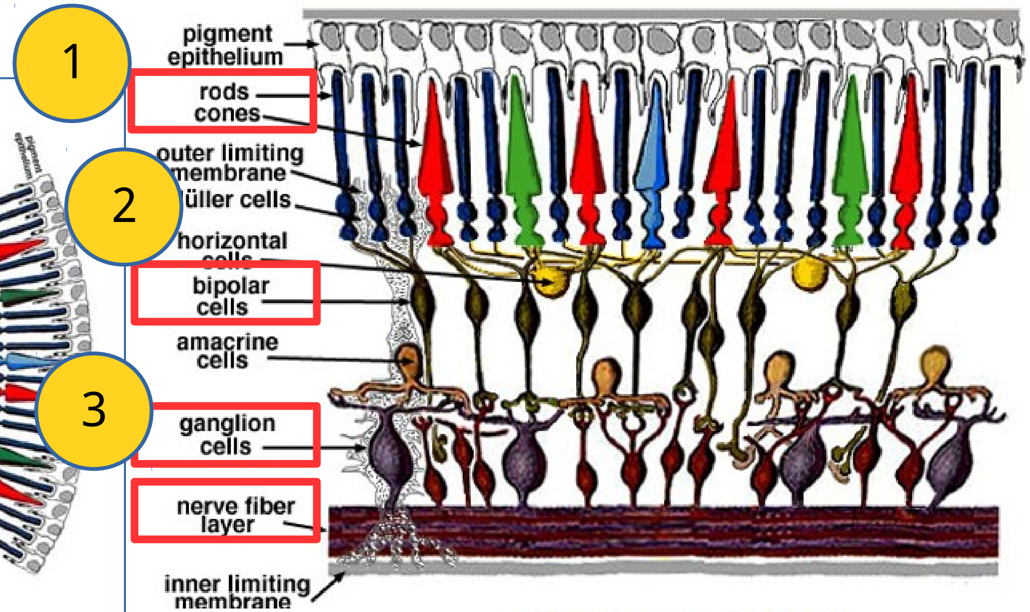
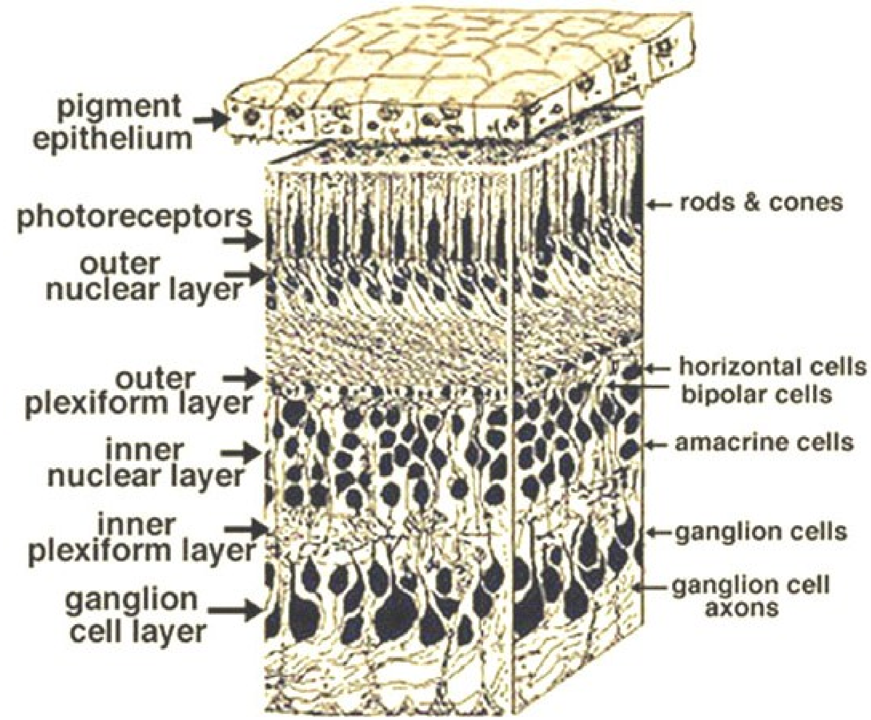


Fig. 1.1. A drawing of a section through the human eye with a schematic enlargement of the retina.

http://webvision.med.utah.edu/imageswv/Sagschem.jpeg

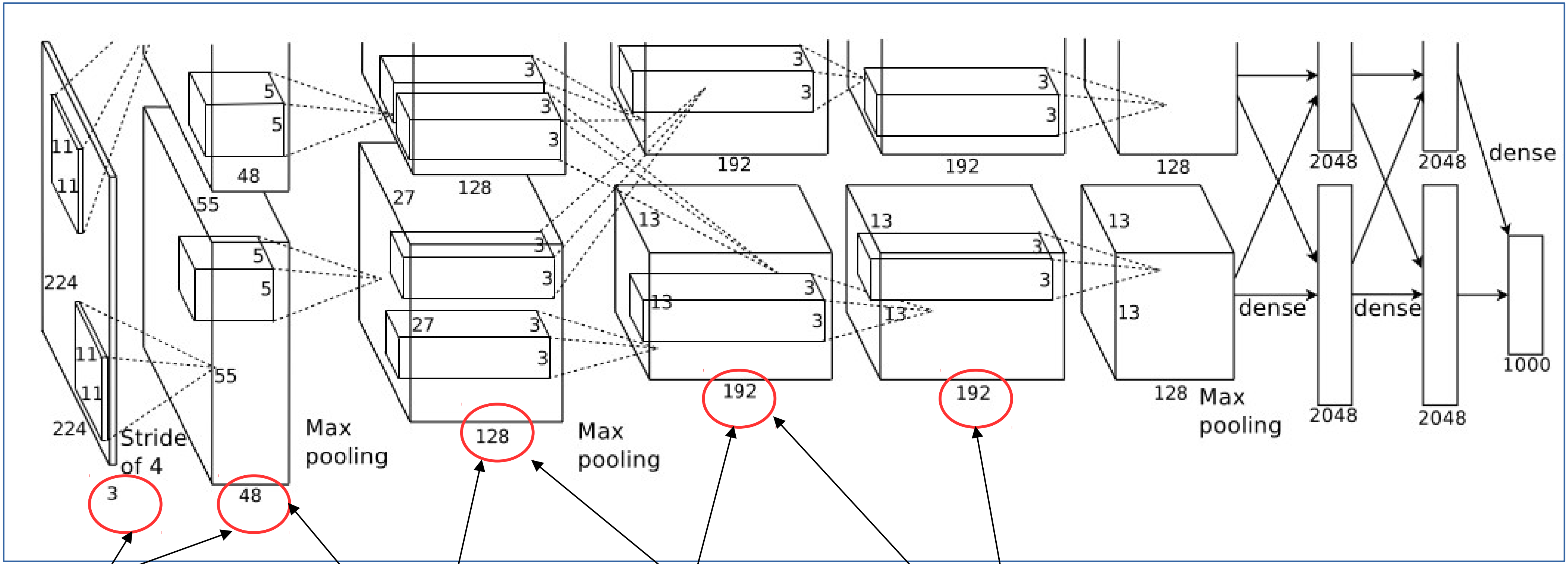




Deep Learning
Alex Net Convolution Layers

Reference: The 9 Deep Learning Papers You Need To Know About (Understanding CNNs Part 3)

https://adeshpande3.github.io/adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html



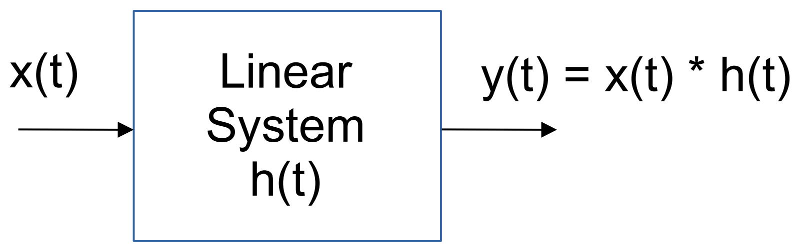
Example 1: 48 kernels, the depth of the kernel is 3

Example 2: 128 kernels, the depth of the kernel is 48

Example 3: 192 kernels, the depth of the kernel is 128

Example 4: 192 kernels, the depth of the kernel is 192

How Is the Convolution Derived



Linear System Definition:
If input x(t) has produces the output y(t), e.g.,

$$\boxed{x(t) \longleftrightarrow y(t)} \qquad \ldots (1)$$

then,

$$\boxed{a\,x1(t) + b\,x2(t) \longleftrightarrow a\,y1(t) + b\,y2(t)} \qquad \ldots (2)$$

Limitation: limited new information can be produced, for example, if input is a (a * x(t) + b), then the output can not be 2^{nd} order or higher, for example no x(t) * x(t) can be produced.

Step 1. Impulse response, delt(t) input, h(t) output;

Step 2. Sampling technique, sample and hold circuit;

Step 3. Nyquest theorem, conversion of continuous system to discrete system

Step 4. Integration of the sequence of all responses by integration .

2D Convolution Computation

Reference for the theoretical background: Chapter 6, Robot Vision, pp. 104 – 111, by BKP Horn, MIT Press

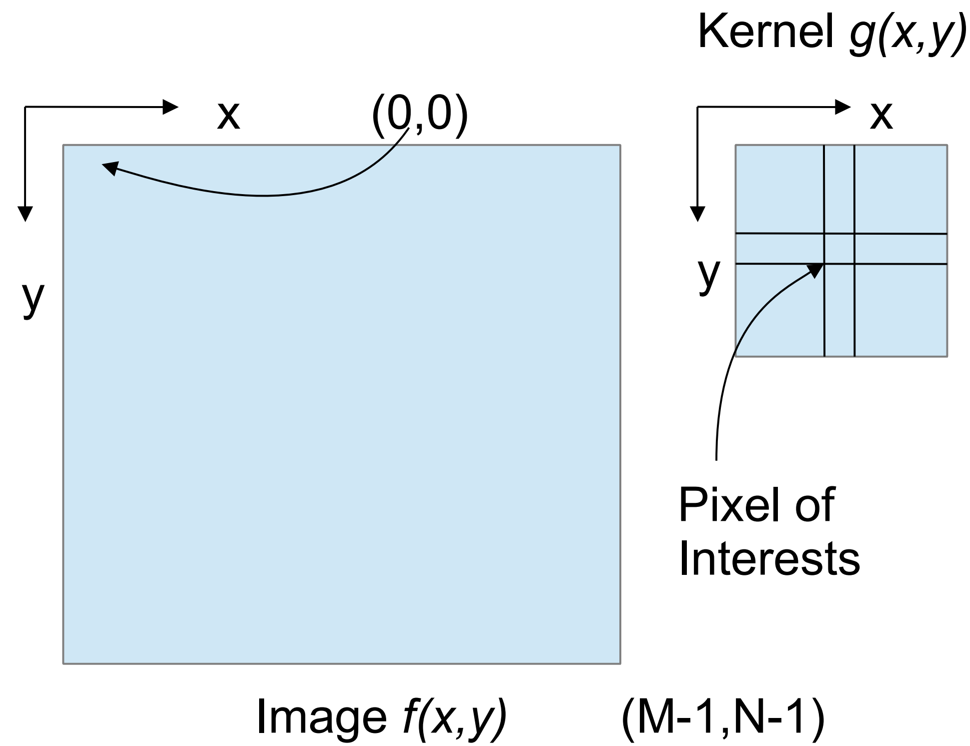
Definition:

$$f(x) * g(x) = \int_{-\infty}^{\infty} f(\tau) \cdot g(x-\tau)\, d\tau$$

$$c(n_1, n_2) = \sum_{\boxed{k_1=-\infty}}^{\boxed{\infty}} \sum_{\boxed{k_2=-\infty}}^{\boxed{\infty}} a(k_1, k_2) b(n_1 - k_1, n_2 - k_2)$$

Image Kernel

Summation lower and upper bound in the case of M-by-N image f(x,y), should be adjusted to k1 = 0 to M-1, k2 = 0 to N-1

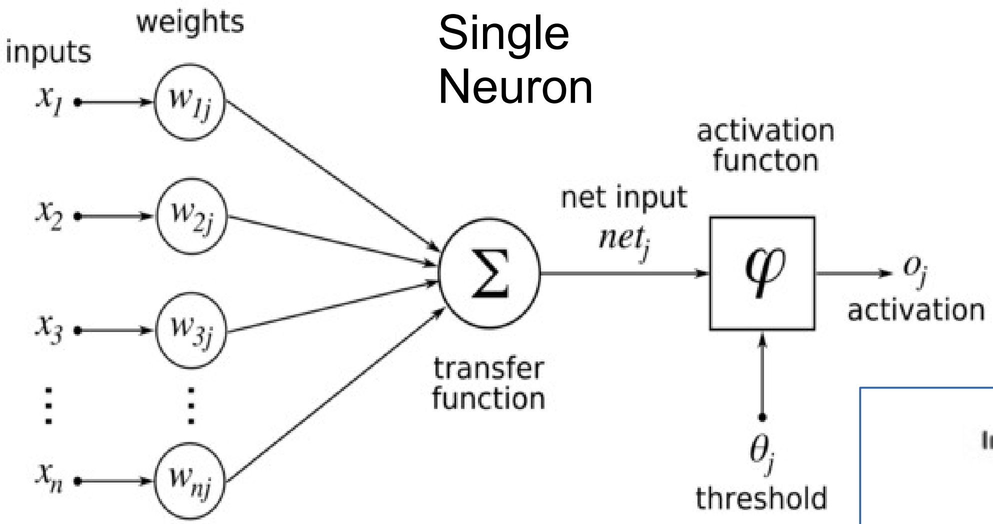


Reference for the OpenCV implementation: Learning OpenCV, Chapter 6, pp. 144 – 164.

Note: (1) 3 primitive computations: shift, multiplication, and addition;
(2) use discrete 2D convolution formula to compute 5x5 sample image with 3x3 kernels

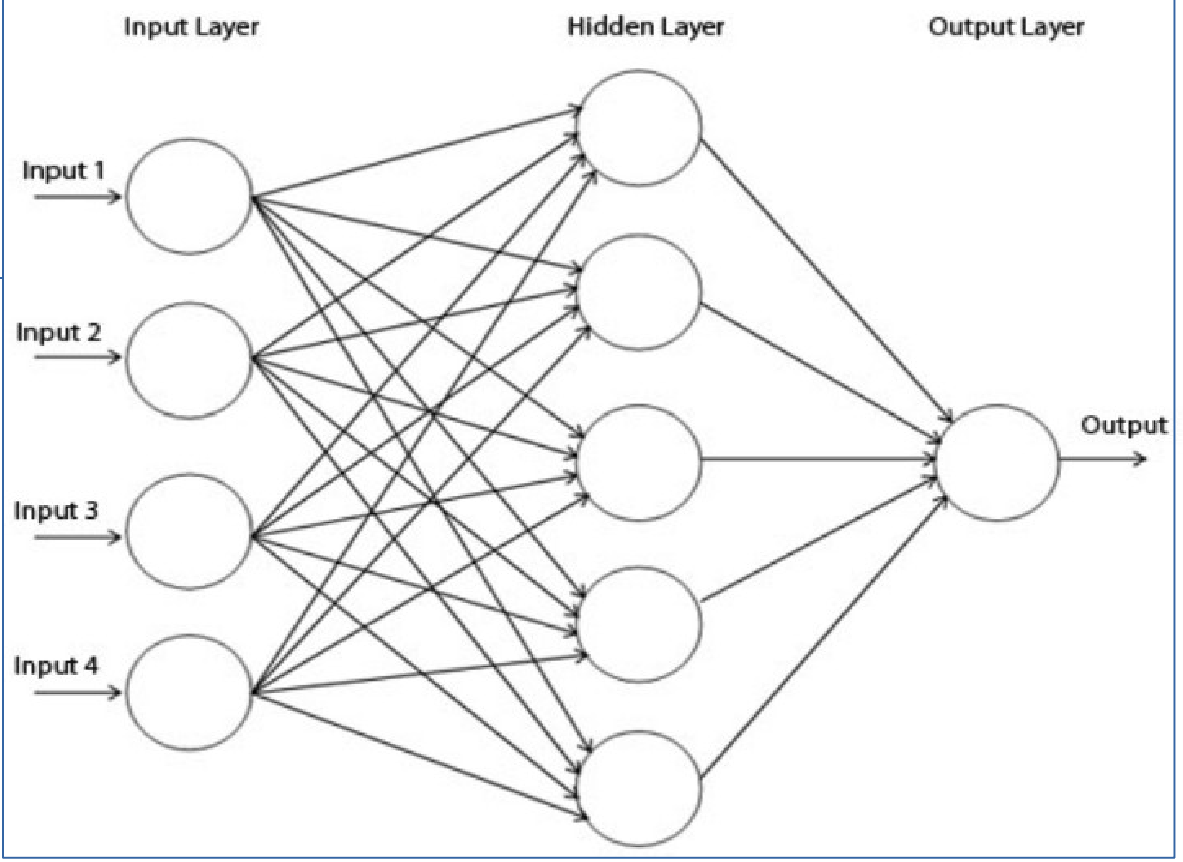
Feed Forward Neural Networks

Single Neuron



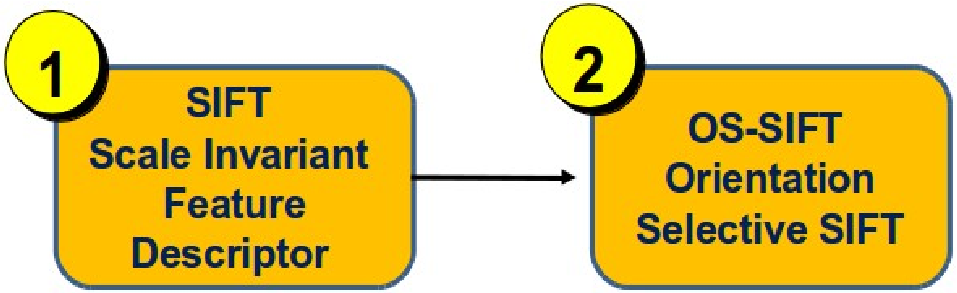
https://d4datascience.wordpress.com/2016/09/29/fbf/

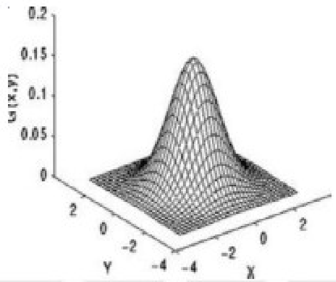
Multi-layer feed forward neural networks

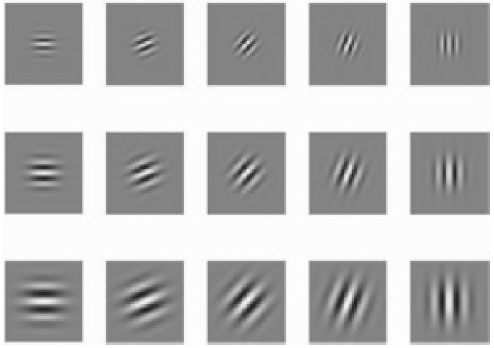


Kernel Mathematical Design

Example:







$$g(x,y;\lambda,\theta,\psi,\sigma,\gamma)=\exp\left(-\frac{x'^2+\gamma^2y'^2}{2\sigma^2}\right)\exp\left(i\left(2\pi\frac{x'}{\lambda}+\psi\right)\right)$$

Kernel Mathematical Design

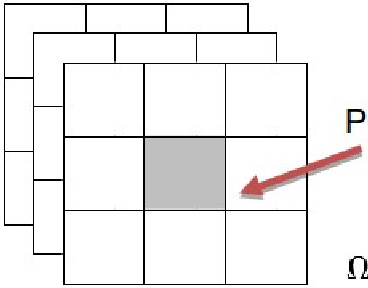
Example:

2 **OS-SIFT**

$$g(x,y;\lambda,\theta,\psi,\sigma,\gamma) = \exp\left(-\frac{x'^2+\gamma^2 y'^2}{2\sigma^2}\right)\exp\left(i\left(2\pi\frac{x'}{\lambda}+\psi\right)\right)$$

Lindberg Function Scale Space image

$Dog(x, y, t; \sigma_i) = [g(x, y; \sigma_i)] * I(x, y, t);$ ... (1)



POI: Pixel of Interests

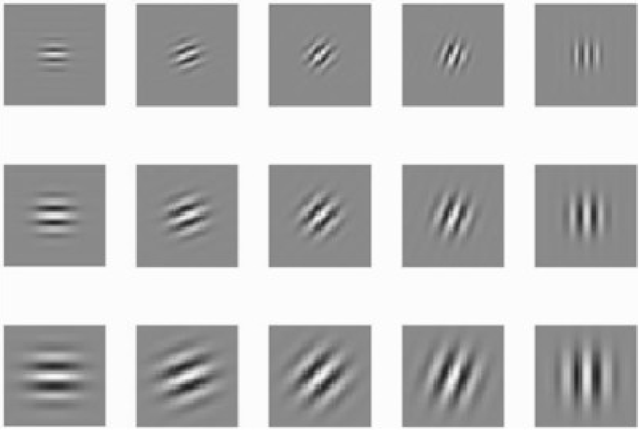
Ω_9 : 3x3 kernel w/POI

DoG Function Difference of Gaussian

$Dog(x, y, t; \sigma_i) = [g(x, y; \sigma_i) - g(x, y; \sigma_{i-1})] * I(x, y, t);$ ... (2)

DoG on 3 KxK kernels Spatial-temporal Characterization

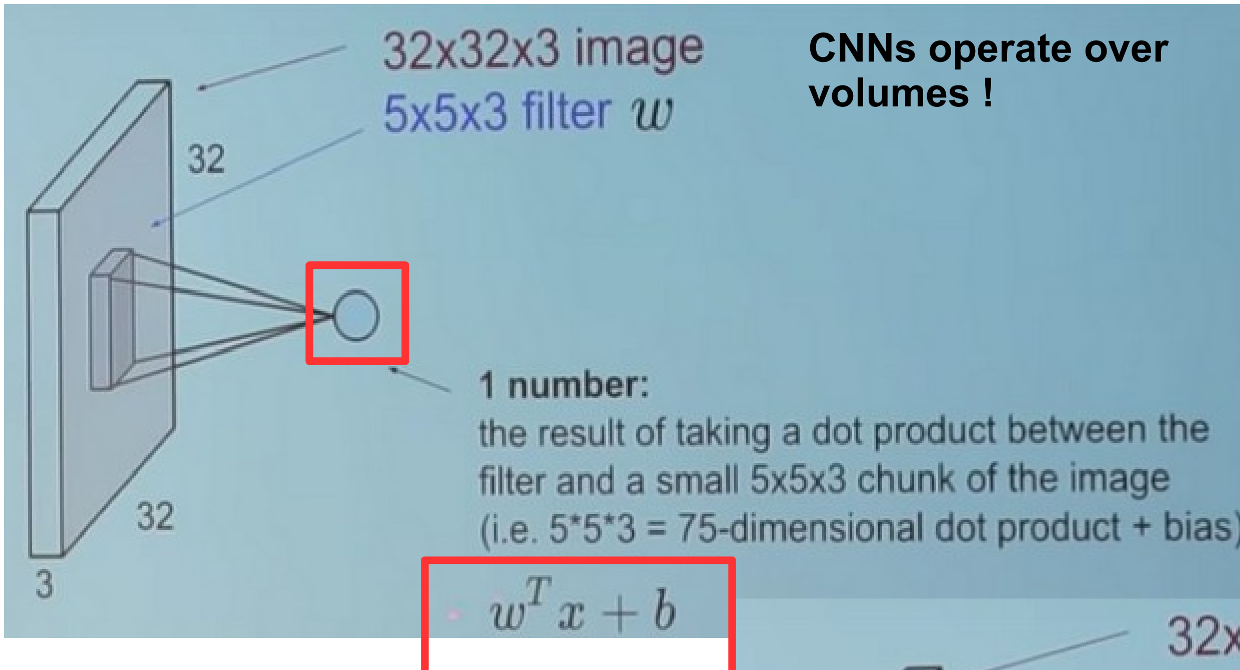
$\{Dog(x, y, t; \sigma_i) | (x, y) \in \Omega_8\}, \{Dog(x, y, t; \sigma_{i+1}) | (x, y) \in \Omega_9\}, \{Dog(x, y, t; \sigma_{i-1}) | (x, y) \in \Omega_9\}$ ... (2)



Feature point descriptor Computation

1. Find $Max\{Dog(x, y, t; \sigma_i)\}\ from\ \sigma_i, \sigma_{i-1}, \sigma_{i+1};$
2. Compute its
   $M(x, y, t; \sigma_i) = sqrt[M_x^2(x, y, t; \sigma_i) + M_y^2(x, y, t; \sigma_i)],$
   $\theta(x, y, t; \sigma_i) = tan^{-1}[M_y^2(x, y, t; \sigma_i) / M_x^2(x, y, t; \sigma_i)];$

Convolution for
Convolutional Neural Netwoks

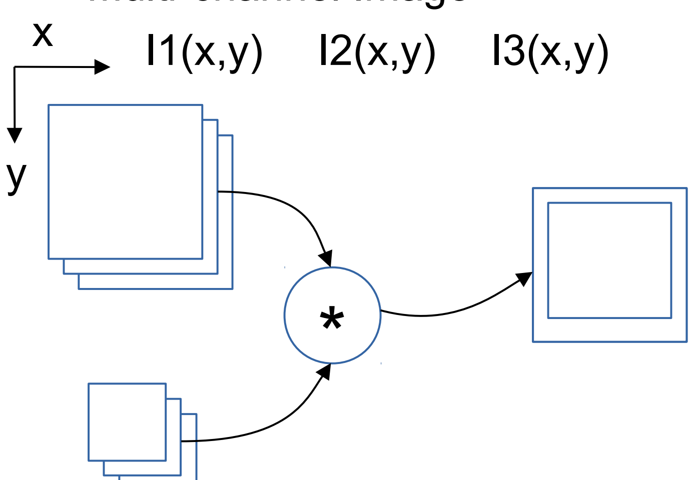


Unlike neural networks, where the input is a vector, here the input is a multi-channeled image (3 channeled in this case).

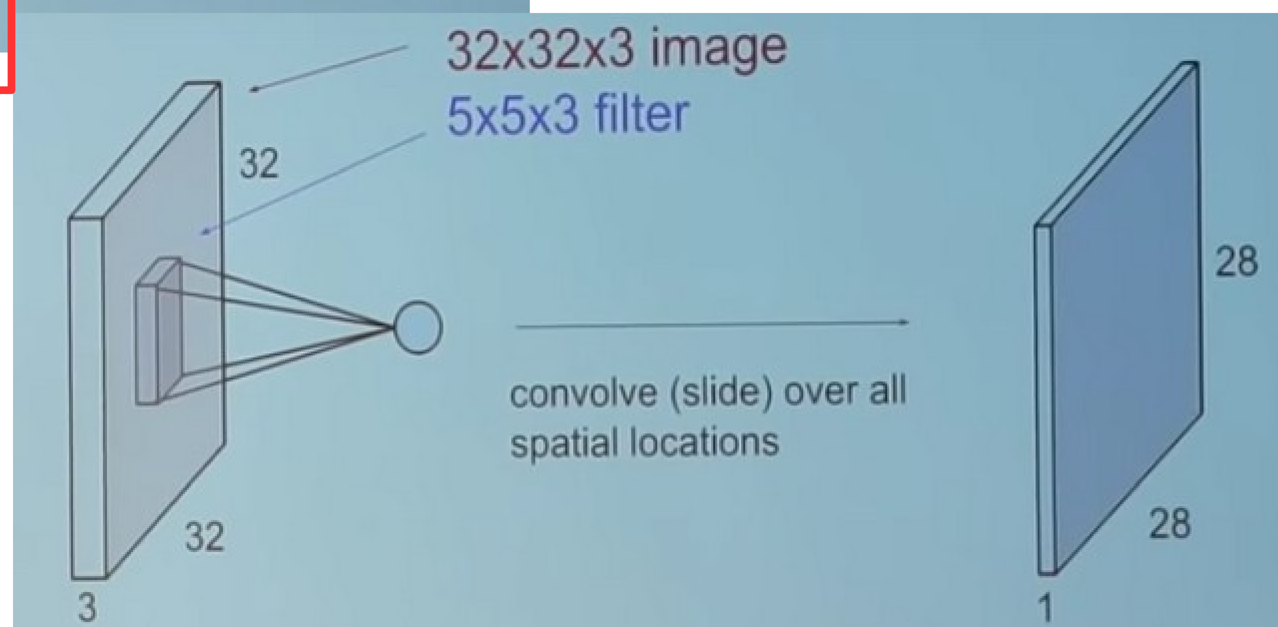
Step 1. Convolution
https://medium.com/technologymadeeasy/the-best-explanation-of-convolutional-neural-networks-on-the-internet-fbb8b1ad5df8

Due to boundary condition, output reduced Size

Multi-channel Image

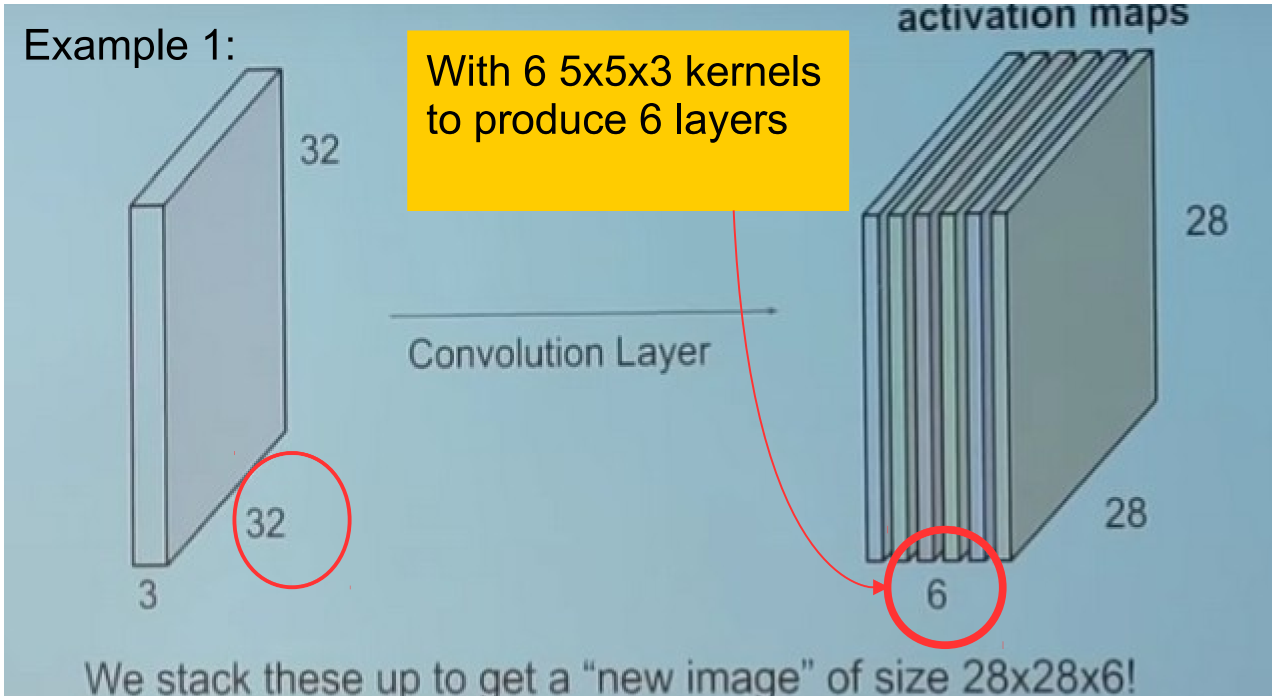


Kernel k(x,y)



Convolution Layers Matching to Number of Kernels

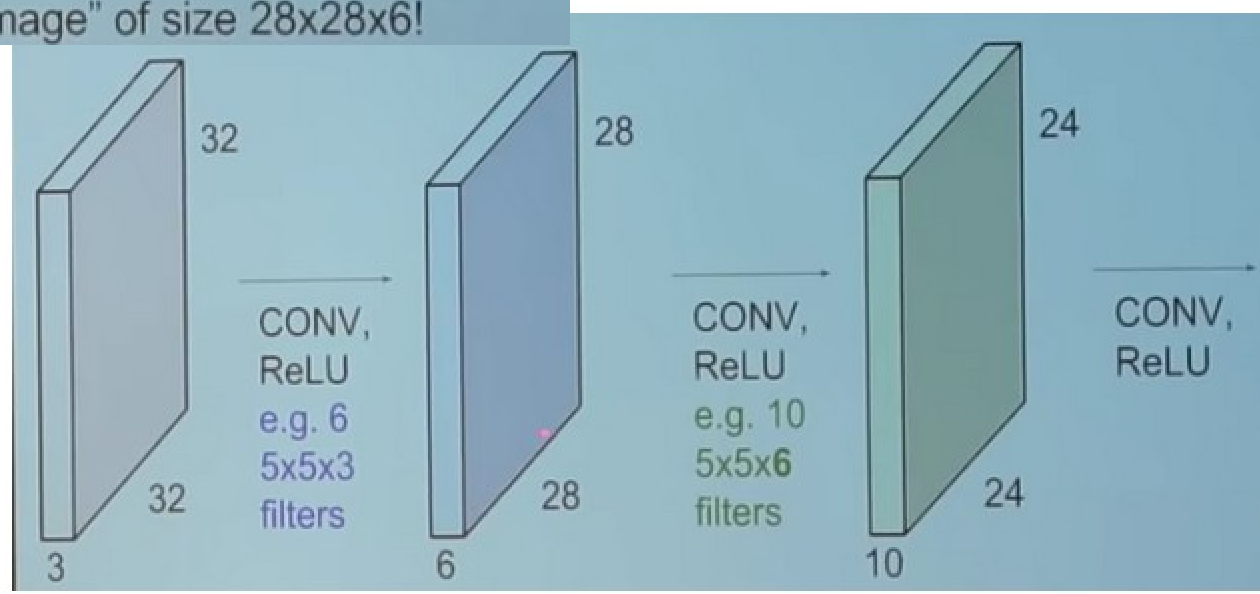
The convolution layer is the main building block of a convolutional neural network.



Observation:

1. The output layers equal to the number of kernels; 2. the depth of the kernel is equal to the number of input layers

The convolution layer comprises of a set of independent filters (6 in the example shown). Each filter is independently convolved with the image and we end up with 6 feature maps of shape 28*28*1.

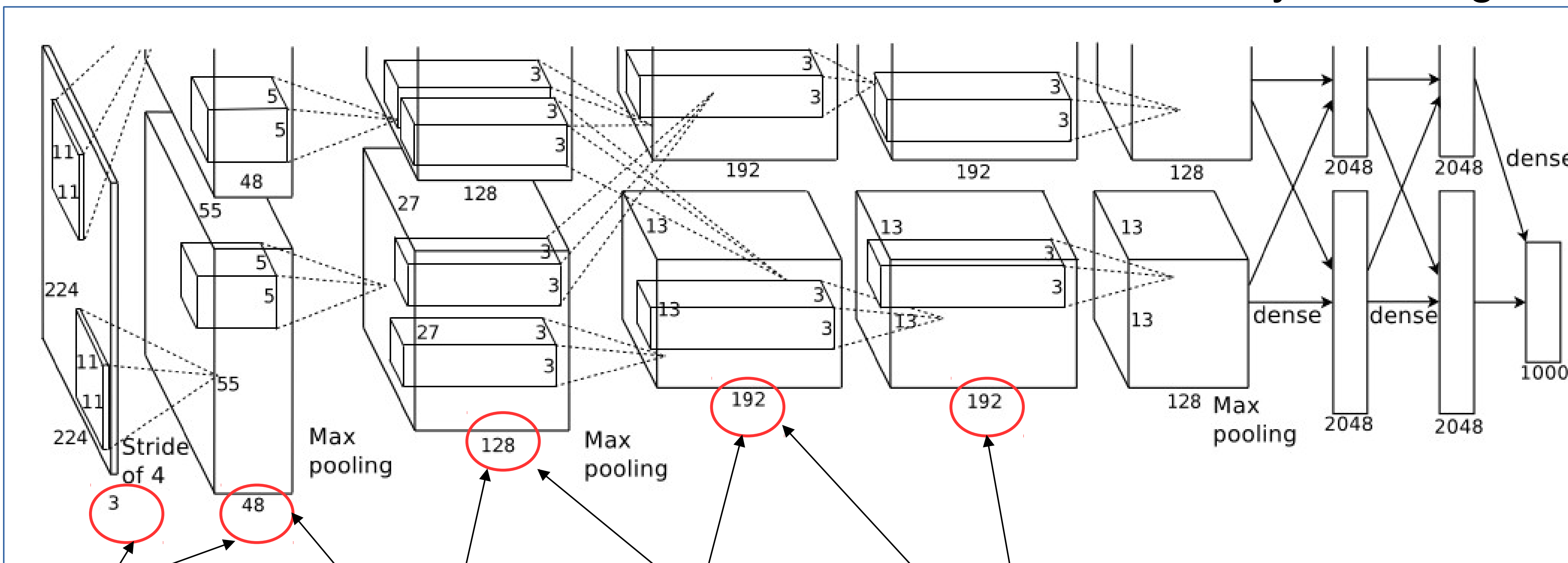


Alex Net Convolution Layers

https://adeshpande3.github.io/adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html

Reference: The 9 Deep Learning Papers You Need To Know About (Understanding CNNs Part 3)

Kernel By Learning



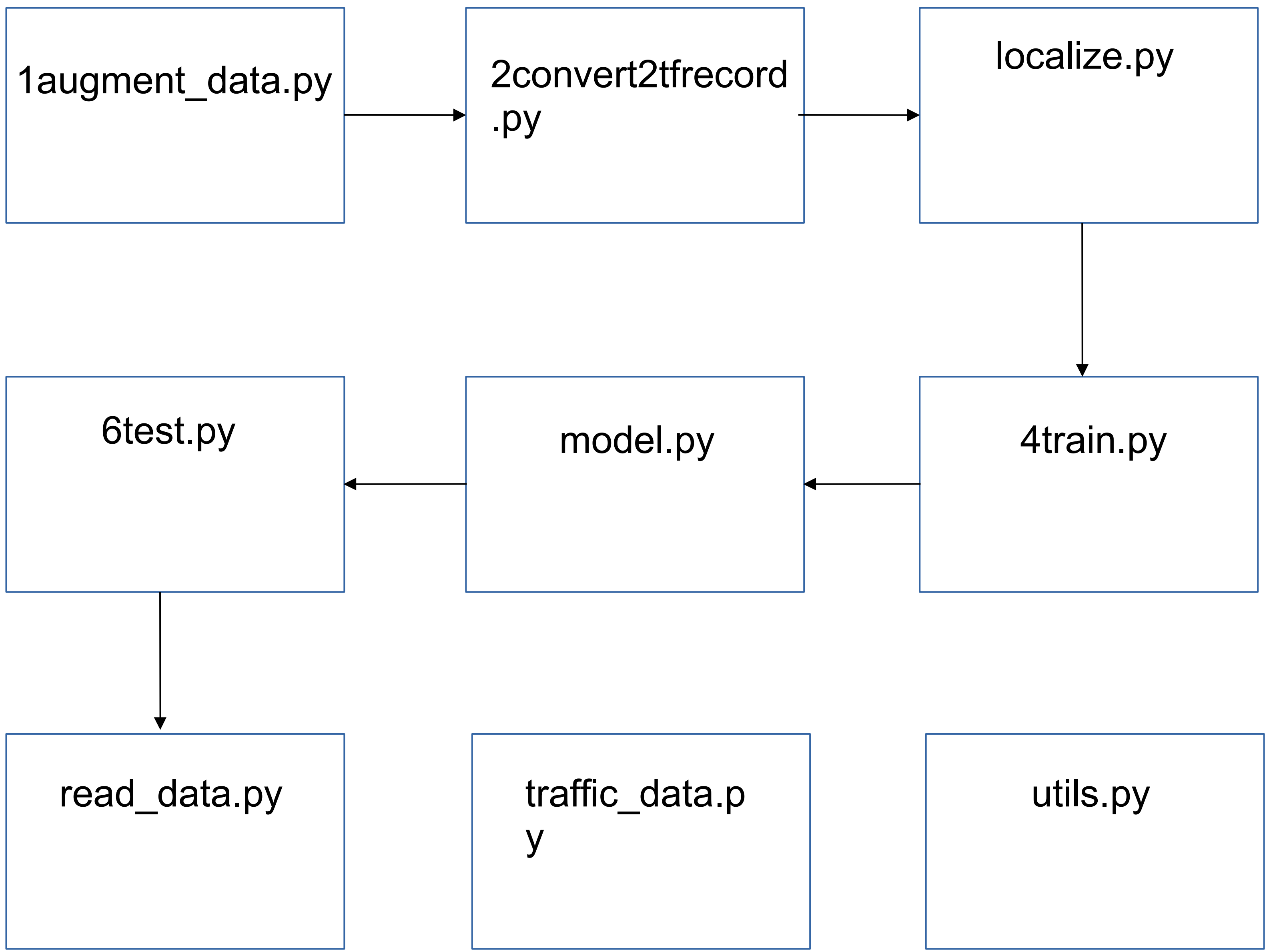
Example 1: 48 kernels, the depth of the kernel is 3

Example 2: 128 kernels, the depth of the kernel is 48

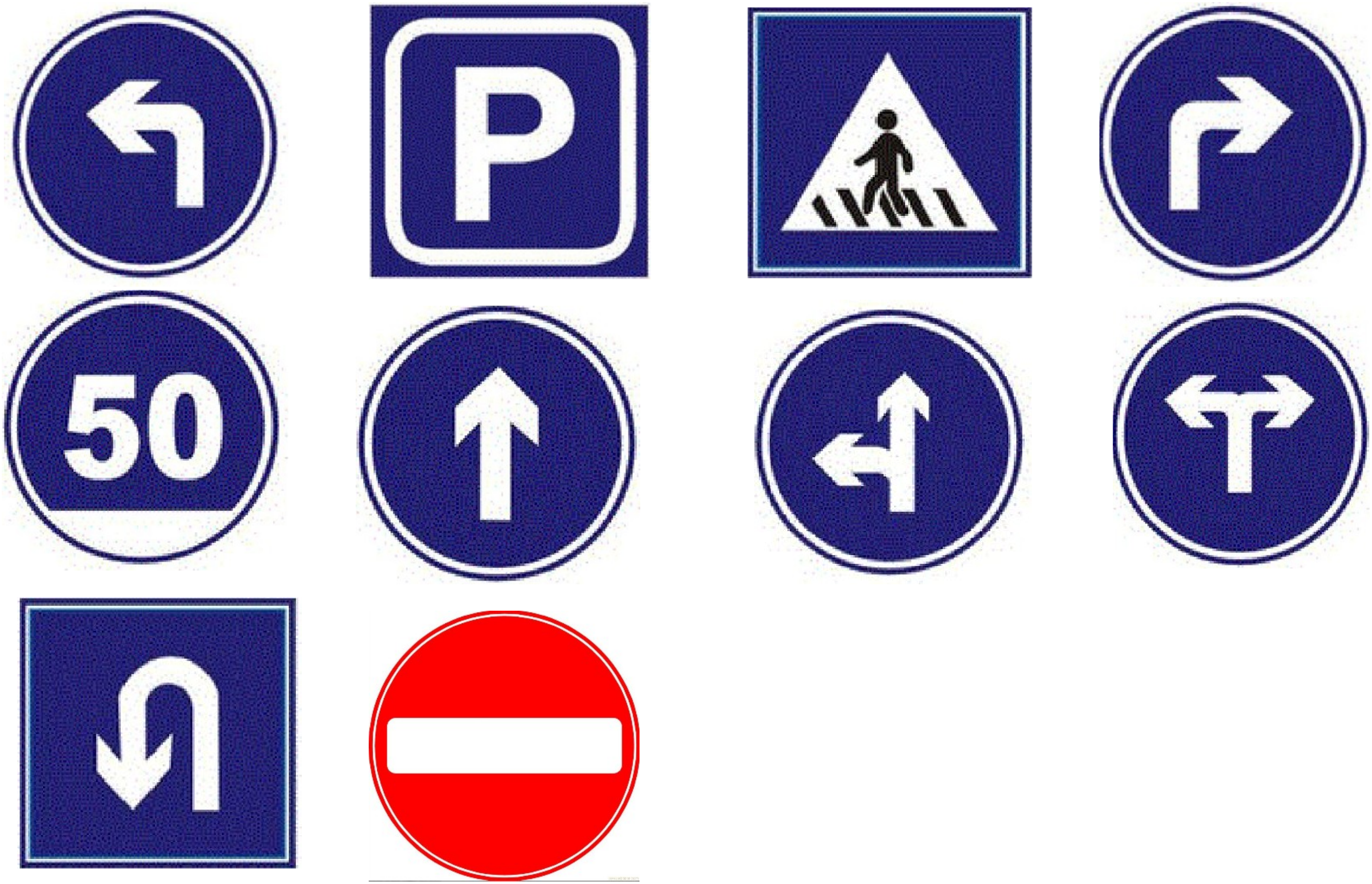
Example 3: 192 kernels, the depth of the kernel is 128

Example 4: 192 kernels, the depth of the kernel is 192

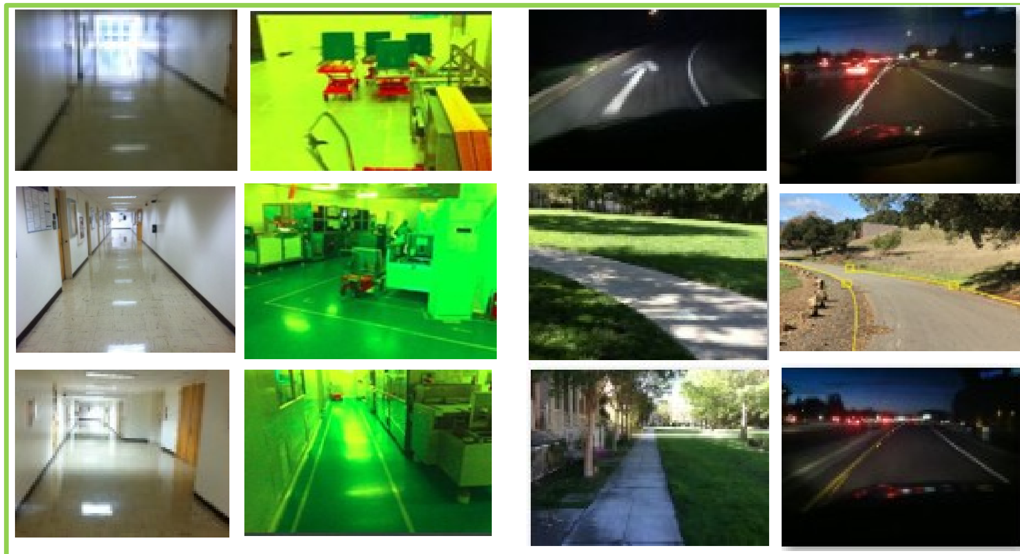
CTI One Production Sample Code



Deep Learning Traffic Signs



Vision Based Motion Path Extraction



Room

Warehouse

P-Walk

High way

Indoor

Outdoor

Big Data with Pre-stored Vision Feature Space

Road Feature Descriptor Extraction

Road boundary extraction

Path Trajectory Planning

Machine Learning for boundary extraction